

# Developing Object Detection, Tracking and Image Mosaicing Algorithms for Visual Surveillance

by

Taygun Kekeç

Submitted to the Graduate School of Sabancı University  
in partial fulfillment of the requirements for the degree of  
Master of Science

Sabancı University

August, 2013

Developing Object Detection, Tracking and Image Mosaicing  
Algorithms for Visual Surveillance

APPROVED BY:

Prof. Dr. Mustafa Ünel  
(Thesis Advisor)



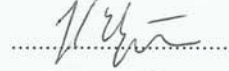
Assoc. Prof. Dr. Ali Koşar



Assoc. Prof. Dr. Erkan Savaş



Assist. Prof. Hakan Erdoğan



Assist. Prof. Dr. Hüseyin Üvet



DATE OF APPROVAL:

05/08/2013



© Taygun Kekeç 2013

All Rights Reserved

# Developing Object Detection, Tracking and Image Mosaicing Algorithms for Visual Surveillance

Taygun Kekeç

ME, Master's Thesis, 2013

Thesis Supervisor: Prof. Dr. Mustafa Ünöl

Keywords: Detection, Tracking, Background Subtraction, Image  
Registration, Stitching and Mosaicing

## **Abstract**

Visual surveillance systems are becoming increasingly important in the last decades due to proliferation of cameras. These systems have been widely used in scientific, commercial and end-user applications where they can store, extract and infer huge amount of information automatically without human help.

In this thesis, we focus on developing object detection, tracking and image mosaicing algorithms for a visual surveillance system. First, we review some real-time object detection algorithms that exploit motion cue and enhance one of them that is suitable for use in dynamic scenes. This algorithm adopts a nonparametric probabilistic model over the whole image and exploits pixel adjacencies to detect foreground regions under even small baseline motion. Then we develop a multiple object tracking algorithm which utilizes this algorithm as its detection step. The algorithm analyzes multiple object interactions in a probabilistic framework using virtual shells to track objects in case of severe occlusions. The final part of the thesis is devoted to an image mosaicing algorithm that stitches ordered images to create a large and visually attractive mosaic for large sequence of images. The proposed mosaicing method eliminates nonlinear optimization techniques with the capability of real-time operation on large datasets. Experimental results show that developed algorithms work quite successfully in dynamic and cluttered environments with real-time performance.

# Görsel Gözetim için Obje Tespiti, Takibi ve Görüntü Mozaikleme Algoritmalarının Geliştirilmesi

Taygun Kekeç

ME, Master Tezi, 2013

Tez Danışmanı: Prof. Dr. Mustafa Ünel

Anahtar Kelimeler: Obje Tespit, Takip, Arkaplan Çıkarma, Görüntü Kayıt,  
Dikme ve Mozaikleme

## Özet

Son yıllarda kameraların ucuzlamasıyla görsel gözetleme sistemlerinin önemi gitgide artmaktadır. Bilimsel, ticari ve son kullanıcı uygulamalarında yaygın olarak kullanılan bu sistemler yoğun miktarda bilgiyi depolayabilir, ayıklayabilir, ve bu bilgileri bir insanın yardımı olmadan yeni bilgi çıkarımında kullanabilir.

Bu tezde, bir görsel gözetim sistemi kapsamında kullanılabilecek obje tespit, takip ve görüntü mozaikleme algoritmalarının geliştirilmesine yoğunlaşmıştır. İlk olarak gerçek zamanlı çalışabilen, hareket ipuçlarını kullanan obje tespit algoritmaları incelenmiş ve dinamik sahnelerde de çalışabilecek bir obje tespit algoritması geliştirilmiştir. Adı geçen algoritma, nonparametrik olasılıksal bir model aracılığı ile piksel komşuluklarını kullanarak görüntüdeki önplan bölgelerini ufak kamera hareketleri altında tespit edebilmektedir. Bundan sonra, önerilen obje tespiti algoritmasını bir önadım olarak kullanan bir çoklu obje takibi yöntemi geliştirilmiştir. Algoritma çoklu obje etkileşimlerini olasılıksal bir çerçevede inceleyerek, sanal kabuklar ile yoğun örtmeye sahip durumlarda objeleri takip etmektedir. Tezin son bölümünde ise sıralı görüntüleri dikerek daha geniş ve görsel olarak etkileyici bir mozaik oluşturan bir görüntü mozaikleme algoritması önerilmiştir. Önerilen yöntem lineer olmayan eniyileme tekniklerini devre dışı bırakarak, geniş görüntü kümeleri için dahi gerçek-zamanlı çalışabilmektedir. Deney sonuçları göstermektedir ki, önerilen algoritmalar gerçek zamanlı olarak dinamik ve kalabalık ortamlarda başarıyla çalışmaktadır.

## Acknowledgements

It is a great pleasure to extend my gratitude to my thesis advisor Prof. Dr. Mustafa Unel, who has taught me patience and contemplation, for his precious guidance and support. I am indebted to him for his supervision and excellent advises throughout my Master study.

I would gratefully thank Assoc. Prof. Dr. Hakan Erdogan for his support through my studies.

Finally, I would like to thank my mother who I can not pay back her efforts for my studies, and to my colleagues Barış Can Üstündağ, Alper Yıldırım, Mehmet Ali Güney, Soner Ulun, Sanem Evren, and my friends Mehmet Fatih Doğan, Salih Yıldırım, Sümeyra Balcı, Sinem Karacabey and Esra Nur Varlı who supported me throughout my studies.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	3
1.2	Thesis Organization and Contributions . . . . .	5
<b>2</b>	<b>Object Detection with Background Subtraction</b>	<b>7</b>
2.1	Real-Time Background Subtraction Algorithms . . . . .	8
2.2	Experimental Results . . . . .	14
2.3	Detection in Dynamic Scenes . . . . .	20
2.4	Implementation Results . . . . .	24
2.5	Discussions . . . . .	25
<b>3</b>	<b>Real-Time Multiple Object Tracking Using Virtual Shells</b>	<b>30</b>
3.1	Background Subtraction . . . . .	32
3.2	Shell Model . . . . .	33
3.3	Event Resolution . . . . .	34
3.4	Pixel Membership Evaluation . . . . .	38
3.5	Position Update . . . . .	40
3.6	Experiments . . . . .	40
3.7	Discussions . . . . .	50
<b>4</b>	<b>Large Scale Mosaic of UAV Image Sequence</b>	<b>51</b>
4.1	Pairwise and Warped Alignment . . . . .	55
4.2	Robust Estimation . . . . .	59
4.3	Multi-Image Alignment . . . . .	61
4.4	Offline Enhancements . . . . .	63
4.5	Experimental Results . . . . .	68

4.6	Discussions . . . . .	68
<b>5</b>	<b>Conclusions and Future Work</b>	<b>72</b>

## List of Figures

1.1	Estimated billion dollar market of video surveillance in 2012-2020 [1]. . . . .	5
2.1	A 2D GMM model having 3 different mixtures each having different mean and variance . . . . .	11
2.2	Rows correspond to original image and results of Adaptive Median, Prati Median and EigenBackground algorithms on car scene. . . . .	16
2.3	Rows correspond to original image Grimson GMM, Zivkovic GMM and Wren GMM algorithms respectively on car scene. .	17
2.4	Precision and recall values of the experiments. . . . .	18
2.5	Result of GMM algorithm on Camera1 sequence obtained from PETS200 database. . . . .	20
2.6	Some scenes having dynamic pixels like tree leaves, flowing water and shiny aquarium pebbles. . . . .	21
2.7	a) An image from aquarium dataset b) Adaptive-Median based background subtraction result c) GMM based background subtraction result d) Proposed background subtraction result. . .	26
2.8	a) Two image from aquarium dataset b) Result of Background and Foreground Modeling c) 3x3 Weighted Median Filter as last step d) 5x5 Weighted Median Filter as last step . . . . .	27
2.9	a) Two image from railroad dataset b) Result of Background and Foreground Modeling c) 3x3 Weighted Median Filter as last step d) 5x5 Weighted Median Filter as last step . . . . .	28

2.10	a) Two image from PETS dataset b) Result of Background and Foreground Modeling c) 3x3 Weighted Median Filter as last step d) 5x5 Weighted Median Filter as last step . . . . .	29
3.1	a)An object $O_i$ with its minimum shell $S_b$ and instantaneous shell $S_r$ is depicted. b) Shells dynamically grow and shrink with time. . . . .	34
3.2	a) State transitions of object to object relations. b) State graph where thin edges represent INTERACTION and thick edges represent JOINT relationship between objects. . . . .	35
3.3	Demonstration of Case IV (green and red) and Case V (purple and orange). . . . .	37
3.4	a) Aquarium image b) Background subtraction c) Output of the event resolution and pixel membership evaluation where pixels of each object are colored differently for illustration purposes . . . . .	39
3.5	Our experimental setup: aquarium environment. . . . .	44
3.6	Sequence from PETS database. . . . .	45
3.7	Tracking accuracy for PETS Sequence where each color represents a tracked object. . . . .	46
3.8	Sequence from Caviar database . . . . .	47
3.9	A tracking sequence from aquarium where three fish interact .	48
3.10	A tracking sequence from aquarium where several fish interact	49
4.1	Flowchart of our mosaicing approach. . . . .	56



4.2	Drift caused by estimation errors. UAV returns to same area and snaps same image from initial position. True and estimated trajectories are shown with green and red dashed curves respectively. . . . .	57
4.3	Comparison of pairwise stitching using Equation 1 and stitching using cost function 3 (bottom). Note that misregistrations caused by error in the leftmost frames are prevented in the second case. . . . .	59
4.4	An illustration of SAT. For a selected projection axis $P_k$ , projected convex sets did not intersect. Dashed green line is an example separator for this projection axis. . . . .	64
4.5	Czyste sequence. Images represent before and after blending respectively. . . . .	69
4.6	Final aerial mosaic constructed from Munich Quarry sequence	70
4.7	Final aerial mosaic constructed from Eastern Island sequence .	70

# Chapter 1

## 1 Introduction

This thesis addresses the problem of developing robust object detection, tracking and mosaicing algorithms in the context of visual surveillance. Visual surveillance systems employ these algorithms as submodules. Nevertheless, these algorithms have diverse application areas and can also be utilized in different contexts. We focus on obtaining effective solutions to aforementioned problems under real time constraints.

Surveillance is monitoring behavior, activities, or other changing information, usually of people for the purpose of managing, directing, or protecting. Surveillance systems are technological tools to increase perception and situational awareness [2]. These automated systems help decision making process of humans with their analysis of events in the scene. As humans, we do not have ability to gather and store high quantity of information. Instead, surveillance systems can gather gigantic quantity of data and process it in which humans cannot perform easily. In literature, there have been different modalities for surveillance systems [3, 4].

There have been surveillance systems based on chemical, sound and visual sensors [5]. Out of all these systems, visual surveillance has proved to be most useful due to proliferation of video cameras and recent advancements in computer vision techniques. With our current technology, the concept of

visual surveillance is mostly implemented using CCD cameras. The ultimate goal of these visual surveillance systems is to extract meaningful information from acquired dense image data. To achieve this purpose, a visual surveillance system consists of several modules such as object detection, tracking and recognition [6]. Detection and tracking algorithms facilitate automatic detection of targets and tracking their behaviour without any human intervention.

Automatic object detection is an attempt to scene understanding and video analysis. It is an essential step to make inference about the scene. These methods can infer number of objects and existence of specific objects in that scene. The methods can exploit object motion [7], appearance [8] or both. These algorithms may be run on each acquired frame, or until a tracking procedure is initialized.

Object tracking is quite popular in many applications. There have been extensive studies on the topic [9]. In literature, much research has been done using point [10], kernel [11] and curve tracking [12]. Many high level applications contain an object tracking procedure. The methods have different object and motion representations based on the application scenario. Apart from selection of appropriate image features and motion models, many of these algorithms require an object detection routine.

Image mosaicing algorithms extend field of view of surveillance system to a much larger scope [13]. The mosaiced image, having a larger field of view, can provide more information than spatially and temporally distinct separate images. These algorithms can be used to stabilize motion or enlarge field of view of a moving camera. In what follows, we focus on finding efficient solutions to these three problems.

## 1.1 Motivation

The problems of object detection, tracking and image mosaicing have captured the interest of both the research and industrial communities in the last decade. These algorithms have such a large applicability that they are commonly used in many different contexts. Surveillance based application areas contain safety in transport applications [14, 15], monitoring of railway stations [16, 17], urban and city roads [18, 19], navigation, tourism and military [20]. Apart from surveillance context, they are also commonly used in fields of medical imagery [21]. The automated systems utilizing these algorithms have noticeably higher response time than human operated systems. Main advantages for using vision based solutions to these problems can be listed as follows:

- They require inexpensive equipment such as a CCD camera and an inexpensive computer.
- They provide high rate data (generally 25-30hz). The upper limit of processing is determined by the power of computing environment.
- The acquisition is passive and more secure unlike other methodologies such as LIDAR.
- Position estimation is also possible without extra sensors.
- Power consumption is lower compared to other sensors.

Nevertheless, the biggest two disadvantages of such vision systems are:

- High vulnerability to visibility conditions such as weather conditions, clouds and other external disturbances.

- Many vision based algorithms require enough visual cues to be found which may not be possible for some scenarios.

In light of these facts, vision based solutions are highly seductive for attacking detection and tracking problems. The systems performing these tasks automatically are becoming more and more popular [2]. The main reason is that even well trained personnel cannot maintain their attention for extended periods of time. They suffer degradation of performance after several hours. Another requirement for such systems is the economical reasons, namely the need for lowering costs. Humans can not maintain multiple tasks simultaneously, one must hire dozens of personal to perform surveillance on multiple or large areas. This increases personal expenses significantly. Moreover, human operator's response time is much slower than automated solutions. All these factors encourage automation of surveillance using robust and effective computer vision algorithms.

These observations are further validated with growing market of automated surveillance. By the year 2012, total revenue in automated surveillance market reached \$13.5 billion. Revenue by the year of 2020 is estimated to reach \$39 billion [1]. The estimated market revenue is shown on Figure 1.1. By year 2011, over 165 million video surveillance cameras installed worldwide captured 1.4 trillion video-hours. Captured video surveillance will reach approximately 3.3 Trillion video-hours in 2020. Deployment of video cameras is so rapidly increasing that even small shops adopt cheap CCTV cameras for security purposes. The market analysis and latest reports about the visual surveillance field concludes that the growth of the field will continue to accelerate.

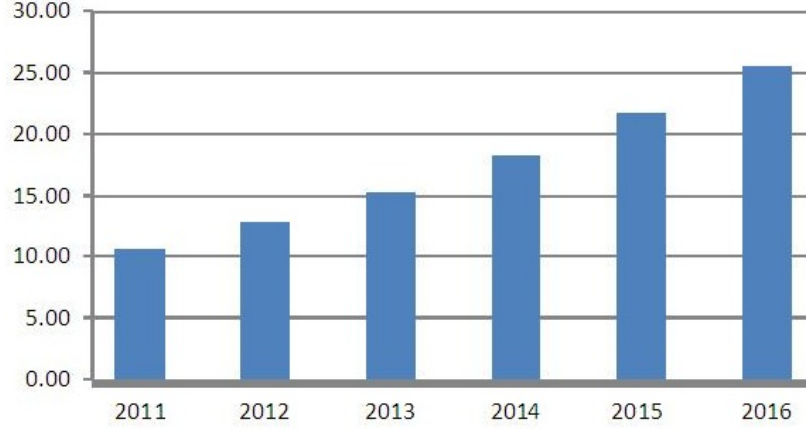


Figure 1.1: Estimated billion dollar market of video surveillance in 2012-2020 [1].

## 1.2 Thesis Organization and Contributions

The purpose of this thesis is to develop effective object detection, tracking and image mosaicing algorithms that can not only be used in surveillance but also in different contexts. In Chapter 2 we first compare some State of the Art object detection methods which are based on background subtraction, and develop an object detection algorithm which can cope with dynamic scenes and robust to camera jitter while being able to operate in real-time speed. The experimental results for proposed method is shown on aquarium setup.

In Chapter 3, we propose a multiple object tracking algorithm that handles object interactions and solves data association problem effectively. The proposed method uses object detection algorithm proposed in Chapter 2 for detecting region of interests in the scene. The tracking results have been shown on aquarium setup and various publicly available datasets.

In Chapter 4, a real-time image mosaicing method has been proposed.

The method uses ordered aerial images captured from a camera under Euclidian motion and capable of creating consistent large scale mosaics in real-time. The novelty of the proposed algorithm lies on avoiding nonlinear minimization while sustaining real-time capability even on large scale data. The experimental results are shown on images captured from an UAV.

Chapter 5 concludes the work done in this thesis and indicates possible future directions.

# Chapter II

## 2 Object Detection with Background Subtraction

Object detection is the process of locating objects in the scene. The methods for detecting objects can be classified into two groups. First group consists of appearance based detection methods [22]. In these methods some visual features from a dictionary is searched in the image. When a viable set of matches is found, an object is meant to be detected [23]. A second group of methods are motion based detection methods [24]. These methods, also called as motion segmentation or background subtraction, exploit motion in the scene to detect abnormalities. The problem can be described as follows: determine each pixel label as foreground or background. Main difficulties of background subtraction are camera jitter, momentary illumination and low quality images due to cheap visual sensors. Moreover, movement of scene objects, new objects added to the scene, object shadows are challenges need to be faced for developing a robust approach. A robust approach must consider all these difficulties under memory and speed constraints [25].

First background subtraction methods were based on creating a subtraction image between consequent images using a binary thresholding mechanism to obtain pixel labels. These methods were fast but not so robust on challenging scenarios and were very sensitive on selected threshold value.



There have been some methods that propose to store a windows of pixel history in time domain and obtain a representative profile of each pixel to belong to the background. The thresholding was carried out after profiling step. Nevertheless, real-time scenes required more advanced methods. The pioneering work in the field has been done by Grimson and Stauffer [26]. They proposed GMM (Gaussian Mixture Model) based background subtraction technique. In the cited work, each pixel was modeled with a number of Gaussian distributions each having different mean and variances. The pixel profile were updated with new incoming pixel values. This probabilistic introduction yielded nice results. Apart from GMM based subtraction, Oliver et al. proposed using Eigen-Space approach [27] for noise-free background subtraction. The whole background image vector was created using a number of learned frames (as samples). Then the vector was averaged and a mean image is formed which is the model for background image.

As the statistical methods rise into power, Kernel Density Estimation based background subtraction approaches become popular [28, 29]. These methods were modeling both background and foreground using a density estimation. They utilize a kernel function (mostly Gaussian or Epanechnikov) to represent data. The most problematic thing with these approaches were memory requirement due to non-discarded data. After background learning phase, collected bandwidth of data represents a nonparametric probability distribution.

## 2.1 Real-Time Background Subtraction Algorithms

In this section, accuracies of several background subtraction algorithms for object detection purposes are evaluated. The selected algorithms have real-

time capabilities. They are learning based where a number of frames must be fed into the learning system in order to learn the background model. After the learning phase, all algorithms create their background model (or background image). The final thresholding is then applied for obtaining pixel classes as foreground or background.

First class of algorithms in this comparison is Median based subtraction algorithms. These algorithms are powerful in the sense that they can cope with arbitrary noise which may be caused by camera jitter or sensor deficiencies. A commonly used algorithm in this class is Adaptive Median [30], which is the fastest and simplest subtraction algorithm in this comparison. After learning a number of sample frames, algorithm computes the median intensity of each pixel in the scene. Using this intensity profile, forthcoming values pixels are evaluated. For each pixel, a history is kept, sorted and median value of the history is updated within a sliding window of frames. Forthcoming pixel value is thresholded to determine whether it is coherent with the background model of that individual pixel, represented with its median.

Another variant of Median based subtraction is proposed by Prati et. al [31]. In the cited work they used a different median function as follows:

$$B^{t+\Delta t}(p) = \arg \min_{\hat{i}} \sum_{j=1}^k D(x_i, x_j), \quad x_i, x_j \in S \quad (1)$$

where  $B^{t+\Delta t}(p)$  is the background model of a pixel at time  $t + \Delta t$ ,  $D$  is the distance function,  $x_i$  and  $x_j$  are elements of that pixel model respectively and  $S$  is the set containing pixel history of background model. The selected minimum argument represents the color model of the pixel. While one can

use different distance functions,  $L_\infty$  distance is chosen in the cited paper:

$$D(x_i, x_j) = \arg \max_c |x_{i,c} - x_{j,c}| \quad (2)$$

where  $c$  denotes selected color in RGB color space. These median based algorithms are computationally more efficient than GMM based subtractions. However, they must be selected only in very limited computing environments in which speed has more importance than detection accuracy.

Another family of detection algorithms are Gaussian Mixture Model (GMM), also called Mixture of Gaussians (MOG), based algorithms. These algorithms have found to be very successful in many scenarios. The first work in this family has been proposed by Stauffer and Grimson [26]. In this approach, each pixel is represented by  $K$  different 1D Gaussian probability densities having distinctive mean and variance (Fig 2.1). Each gaussian density has different weights. In many GMM based algorithm, the covariance matrix is assumed to be diagonal for computational reasons. Given mixture model, probability of a pixel  $X$  at time  $t$  to belong to the background can be given as:

$$P(X_t) = \sum_{i=1}^K w_{i,t} \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \quad (3)$$

where  $P(X_t)$  is the probability of a pixel  $X$  at time  $t$  belonging to the background,  $\eta$  is the normal distribution and  $w_{i,t}$  is the mixture weight of  $i$ th distribution at time  $t$ . Intuitively, recently observed values of each pixel in the scene is characterized by Gaussian mixtures. A new pixel value will be represented by one of the major components of the mixture model and used to update the model. If input pixel intensity is unlikely to fit any model, it

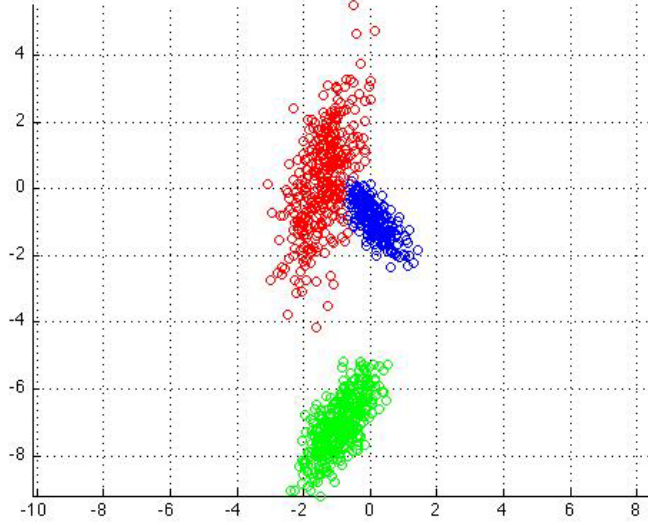


Figure 2.1: A 2D GMM model having 3 different mixtures each having different mean and variance

is considered as foreground.

As background representation of a scene can change in time, each distribution must be updated in time. A common way to update parameters of the probability distributions is EM (Expectation Maximization) algorithm [32]. However to improve real-time capabilities, EM updates can be approximated with K-Means algorithm. In update step, every new pixel value is checked against existing  $K$  Gaussian distributions, until a likely match is found. A match is defined as a pixel value within 2.5 standard deviations of a distribution in the original paper. If none of the  $K$  distributions match the current pixel value, the least probable distribution is replaced with a distribution with the current value as its mean value, an initially high variance, and low prior weight. The weights of  $K$  distributions are updated as a linear

combination with update parameter  $\alpha$ :

$$w_{k,t} = (1 - \alpha)w_{k,t-1} + \alpha(M_{k,t}) \quad (4)$$

where  $w_{k,t}$  is the weight of distribution  $k$  at time  $t$ ,  $\alpha$  is the learning rate and  $M_{k,t}$  is the indicator function for matched distribution. The weights are normalized at each step. Distribution parameters of unmatched distributions remain the same. The parameters of the distribution which matches the new observation are updated as follows:

$$\mu_t = (1 - \rho)\mu_{t-1} + \rho X_t \quad (5)$$

$$\sigma_t^2 = (1 - \rho)\sigma_{t-1}^2 + \rho(X_t - \mu_t)^T(X_t - \mu_t) \quad (6)$$

$$\rho = \alpha\eta(X_t|\mu_k, \sigma_k) \quad (7)$$

where  $\rho$  acts like a causal low-pass filter on the parameter update and  $\alpha$  is the learning parameter. Using  $K$  distributions, GMM has strong aspect dealing with multimodal backgrounds. When some pixel joins the background model, it doesn't destroy whole background model, instead it replaces the weakest representative mixture model. As a last step,  $B$  of the  $K$  distributions are chosen as background model using the equation:

$$B = \arg \min_b \left( \sum_{k=1}^b w_k > T \right) \quad (8)$$

where  $T$  is user-defined tolerance threshold. If  $T$  is higher, a multi-modal distribution caused by a repetitive background motion (leaves, flags) could result in more than one color being included in the background model.

One of the problematic aspect of original GMM based approach [26] is to

choose how many mixture models we need to use. To overcome this problem, Zivkovic [33] proposed an automatic way to choose number of mixture models used. The formulation allows creating new mixtures or combining occurant mixtures into one mixture. They select priors of distributions using Minimum Message Length criteria and then obtain a MAP solution to number of mixture models using Euler-Lagrange equations.

GMM has been widely studied and many extensions to the original algorithm has been proposed such as Wren’s work [34]. The most recent algorithms developed over GMM framework can be found in [35].

Another class of algorithms for background subtraction is EigenBackgrounds. In these approaches, background model is created by first gathering  $N$  sample images  $I_1, I_2, I_3, \dots, I_N$ . Then their mean background image  $M_b$  and covariance matrix  $C_B$  is formed. As size of covariance matrix will be extremely big, the diagonalization of covariance matrix is obligatory. This is done by using an eigenvalue decomposition as follows:

$$L_B = \Phi_B C_B \Phi_B^T \quad (9)$$

where  $\Phi_B$  is the eigenvector matrix of the covariance of the data and  $L_B$  is the corresponding diagonal matrix of its eigenvalues. In order to reduce the dimensionality of the space, only  $M$  eigenvectors out of  $N$  are kept using PCA (Principal Component Analysis). Largest  $M$  eigenvalues are stored in  $L_M$  and the  $M$  vectors corresponding to these  $M$  largest eigenvalues in the matrix  $\Phi_M$ . Once the eigenbackground images are stored, background image  $I_t$  can be represented by the mean background image and weighted sum of the eigenbackgrounds,  $\Phi_M$ . The coordinate in eigenbackground space

of input image  $I_t$  can be computed as follows:

$$W_t = (I_t - \mu_B)^T \Phi_M \quad (10)$$

This is followed with back projection of  $W$  on the image space. A reconstructed background model image  $B_t$  is created as follows:

$$B_t = \Phi_M W_t^T + \mu_B \quad (11)$$

The final thresholding  $|I_t - B_t| > T$  at time  $t$  gives background and foreground pixels.

## 2.2 Experimental Results

The experiments have been conducted on railroad scene [36] having approximately 500 frames. Experimentally it is observed that 200 learning frames are sufficient for all algorithms. The dataset contains noticeable camera jitter and two moving objects. After several seconds one person from the right border of the camera's FOV, and one vehicle from left border of the camera FOV enter to the scene. The color profile of railroad is very similar with moving person's clothing.

The algorithms try to capture motions of these objects. The visual comparison of the algorithms are shown in Figure 2.2 and 2.3. Results show that GMM captures the motion of car and person slightly better than other techniques. Prati-Median (P-Median) and Adaptive Median (A-Median) algorithms show very similar results while being nearly 1.5 times faster than GMM based algorithms. Visually, EigenBackground based background sub-

traction has noticeably better than Zivkovic and Wren GMM variants. It captured most of the pixels of moving car as foreground, however could not eliminate all jittering noise in the background. It should be noted that all algorithms, especially EigenBackground, suffer from the object shadows which is a great challenge for all these algorithms. It is extremely hard to distinguish these shadows without using extra cues or an external mechanism. The shadows of car and person are misclassified as foreground pixels. That is depicted in Fig 2.3.

The quantitative performance of these algorithms has been evaluated using ground truth data of [36]. The ground truth consists of each frame having its background and foreground pixels labeled. For this dataset, precision and recall values can be defined as [28]

$$Precision = \frac{\# \text{ of true positives detected}}{\text{total } \# \text{ of positives detected}} \quad (12)$$

$$Recall = \frac{\# \text{ of true positives detected}}{\text{total } \# \text{ of true positives}} \quad (13)$$

The precision chart in Fig 2.4(a) shows that P-Median and A-Median algorithms has higher true positives than others. This is especially observed in frames 325-500 where a person and a car are moving towards each other. The precision chart depicts that Median based approaches suppresses more pixels which belongs to foreground while Gaussian Mixture based background subtraction algorithms accept them as foreground. This is especially undesired in object detection applications because noisy small misdetections can be favored over losing pixels from foreground objects, which would cause incomplete object representations.



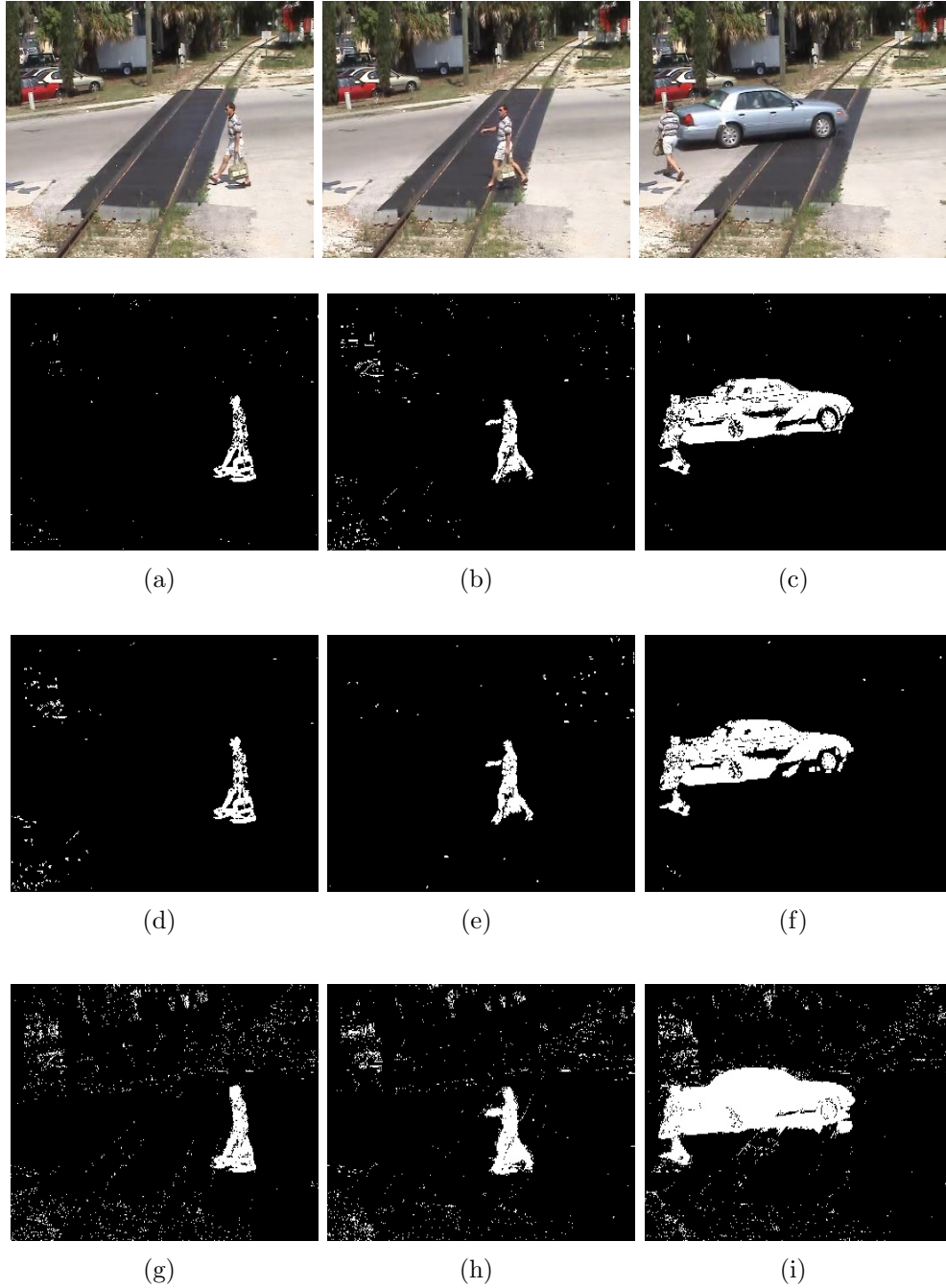


Figure 2.2: Rows correspond to original image and results of Adaptive Median, Prati Median and EigenBackground algorithms on car scene.

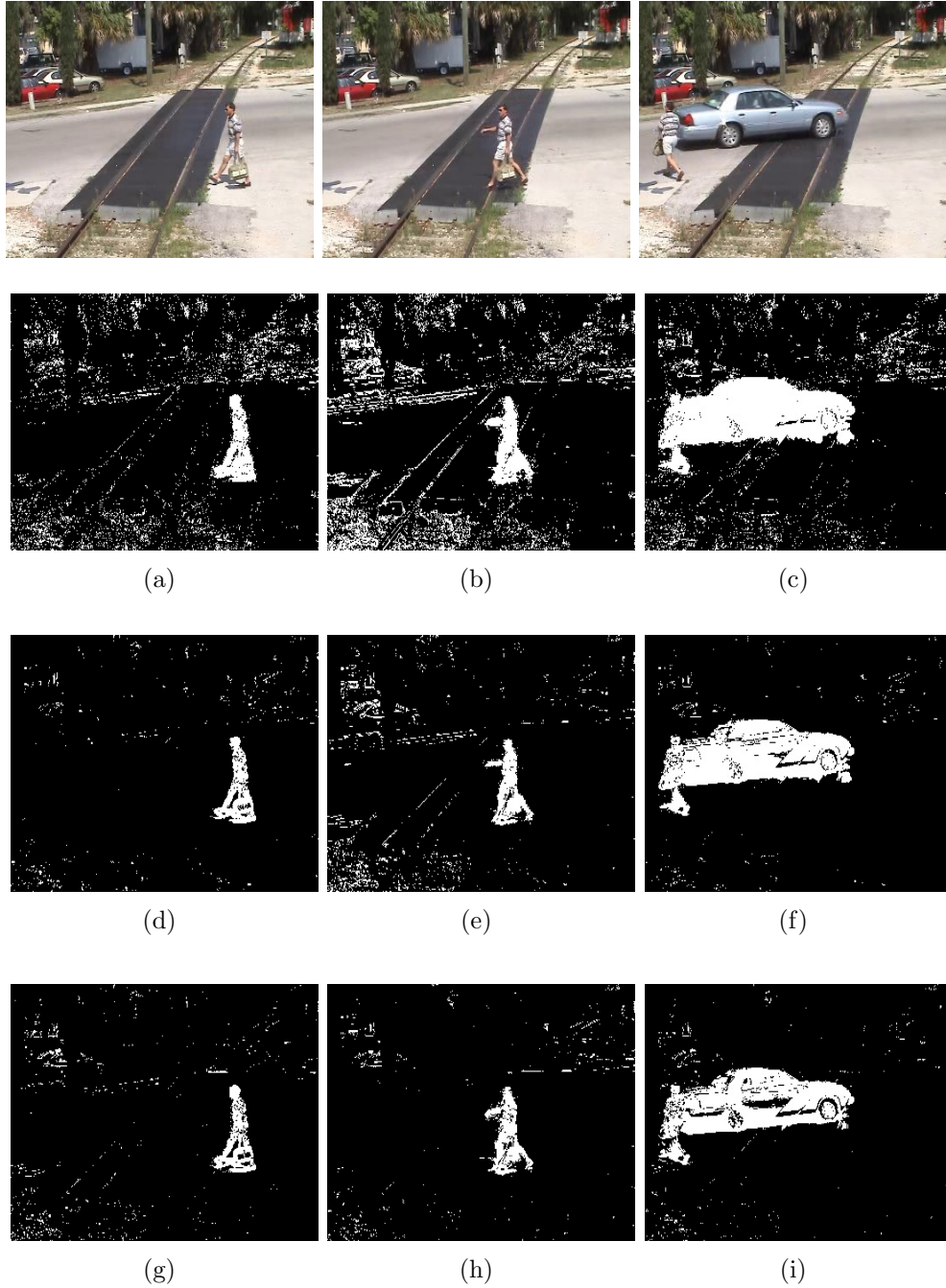
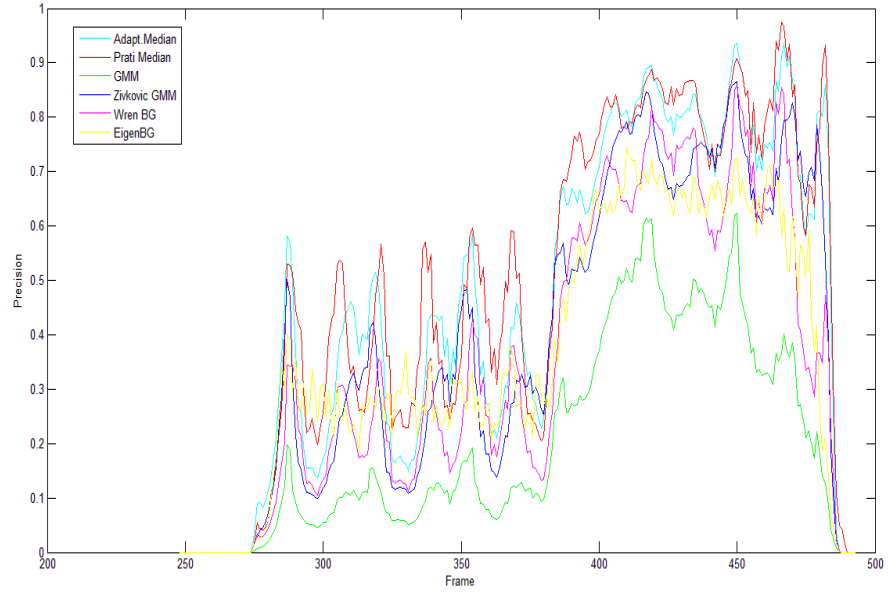
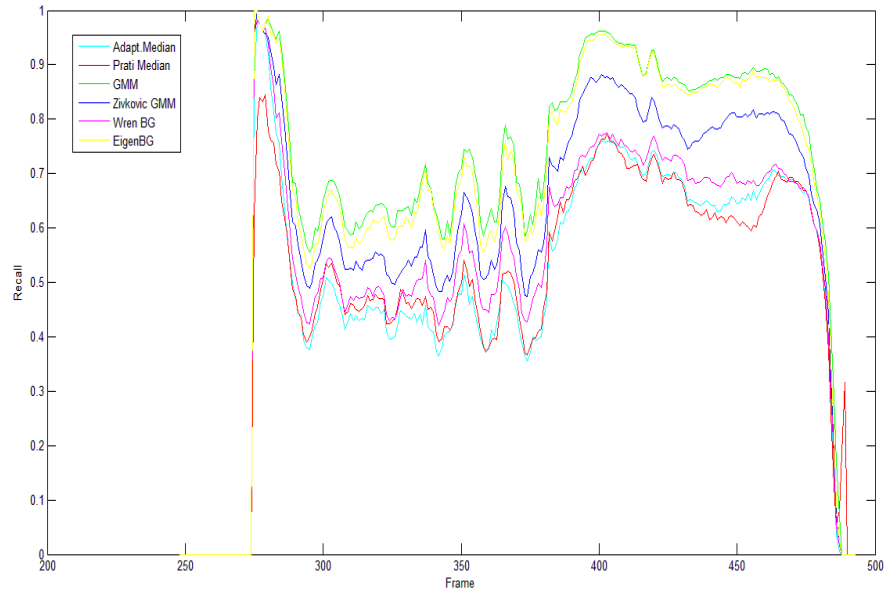


Figure 2.3: Rows correspond to original image Grimson GMM, Zivkovic GMM and Wren GMM algorithms respectively on car scene.



(a)



(b)

Figure 2.4: Precision and recall values of the experiments.

The recall values of GMM, Z-GMM and EigenBackgrounds algorithms are noticeably higher than others as shown in Figure 2.4(b). When both precision and recall values are taken into account, one can comment that GMM based background subtraction algorithms found more foreground pixels than it should, containing more noise than Median based approaches. Median approaches give more reliable foreground information in the data by sacrificing some low reliable foreground pixels to the background. Eigenbackgrounds algorithm shows good overall accuracy. Nevertheless the weakness of Eigenbackgrounds algorithm is that generation of a mean image for background model is computationally expensive especially for high resolution images. Nevertheless, GMM based approaches create background model in no time with K-Means parameter updates.

Conclusion of the comparative results is that Z-GMM performs superior results than all other algorithms in varying real-time conditions. However, while it performs very robust subtraction results on static scenes as shown in Figure 2.5, it can not cope with dynamic scenes effectively due to small jitters due to camera motion. The core assumption of GMM based model namely the value of each pixel is independent of its neighbours is violated in some scenarios. This encourages that either a motion stabilization algorithm needs to be utilized before running GMM algorithm or a model which takes neighboring pixel dependencies into account must be considered. The next section describes a method for obtaining background subtraction algorithm for dynamic scenes having significant camera jitter while preserving real time performance.

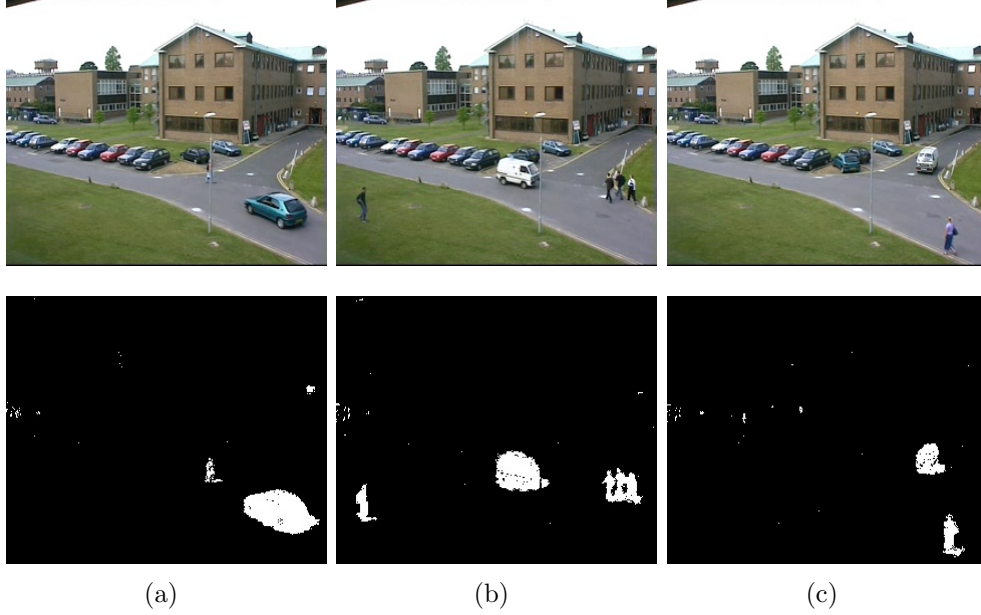


Figure 2.5: Result of GMM algorithm on Camera1 sequence obtained from PETS200 database.

### 2.3 Detection in Dynamic Scenes

In previous section several background subtraction algorithms have been evaluated for real world scenes. Work of [33] showed robust and middle-way performance over other methods. Nevertheless it lacks handling dynamic scenes (see Figure 2.6). For example, an aquarium having pebbles and lighting changes can be considered as a dynamic scene where not choosing appropriate number of mixture model would result in false detection results. Moreover, the same algorithm assumes there is no camera jitter. In order to overcome these two issues, we approach detection problem in dynamic scenes using nonparametric background modeling. The work described here is based on the work of Sheikh [37].

First, each pixel in the image is modeled by a joint feature vector  $\mathbf{x} =$

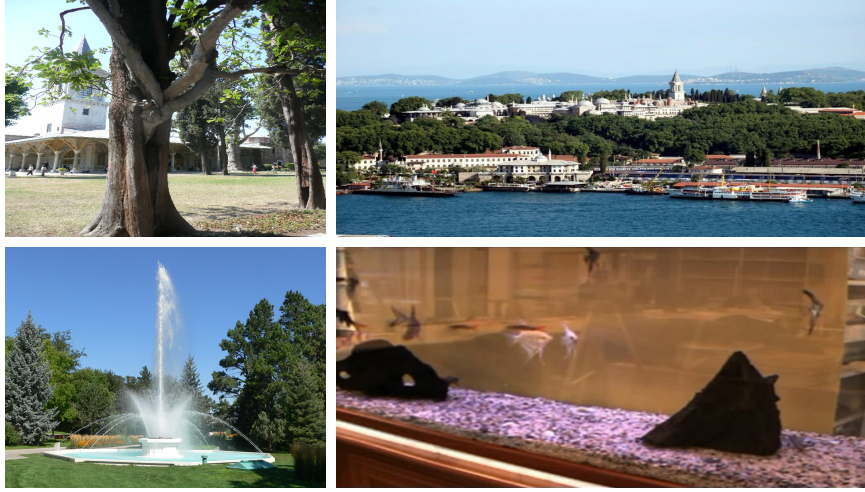


Figure 2.6: Some scenes having dynamic pixels like tree leaves, flowing water and shiny aquarium pebbles.

$(r, g, b, x, y)$ . Such a joint modeling allows sharing of color profiles of neighbour pixels by taking account of spatial dependency. The joint feature vector is used to create a background model:

$$P(x|\psi_b) = n^{-1} \sum_{i=1}^n \varphi_H(\mathbf{x} - y_i) \quad (14)$$

where each candidate  $\mathbf{x}$  pixel is compared to all other pixels in the background model using  $\varphi_H$  kernel function in order to obtain the probability of being a member of the background model. In the next step of the algorithm, a foreground model is constructed assuming that object's position can not have a sudden change and most of the foreground pixels will stay as foreground in the next step. The foreground model also uses same kernel function,  $\varphi_H$ . The formulation for foreground model can be given as:

$$P(x|\psi_f) = \alpha\gamma + (1-a)m^{-1} \sum_{i=1}^m \varphi_H(\mathbf{x} - z_i) \quad (15)$$

where each candidate pixel's foreground membership probability is a mixture of a uniform  $\gamma$  function and a kernel density function  $\varphi_H$ . The uniform function allows new pixels which had never been encountered in background model to be added to foreground model. After obtaining model membership probabilities of a candidate pixel, the probabilities are evaluated under a Parzen classifier function:

$$\tau = -\ln \frac{P(x|\psi_b)}{P(x|\psi_f)} = -\ln \frac{n^{-1} \sum_{i=1}^n \varphi_H(x - y_i)}{\alpha\gamma + (1-a)m^{-1} \sum_{i=1}^m \varphi_H(x - z_i)} \quad (16)$$

After computing  $\tau$  values of all pixels, the final step of subtraction is basic thresholding using  $\delta$  function:

$$\delta(x) = \begin{cases} -1 & \text{if } -\ln \frac{P(x|\psi_b)}{P(x|\psi_f)} > \kappa \\ 1 & \text{otherwise} \end{cases} \quad (17)$$

The thresholded binary image represents background and foreground pixels of that frame. In each frame of the algorithm, all pixels are added to the background model but only foreground labeled pixels are added to the background model. Using bimodal representation for background subtraction greatly increases variance between background and foreground distributions. Selecting one threshold using one background model is harder than selecting one threshold using one background model and a foreground model.

It is obligatory to apply a post-processing operation on the image after determining foreground pixels. The reason behind this is that due to sensorial noise or sudden illuminance change the dependency between neighbourhood

pixels can be lost resulting in misclassified pixels in the model. In literature, overcoming such a noise is possible using morphological operations, filters or global optimization over whole image using Markovian modeling. In the cited work [37] authors claimed that neighbourhood pixel dependencies can be viewed as Markov Random Field obeying Ising model and the formulation of this global optimization problem can be solved using a Graph-Cut algorithm [38], by relying on the fact that graph based energy minimization yields global minimum on applications having two states. In their approach,  $\tau$  values obtained from Parzen classifier is fed into Graph-Cut algorithm as sink and terminal values.

While that formulation achieves optimal minimum and gets rid of many noise pixels in binary images, it is known that global optimization using GraphCut formulation is far from being real time. Using an already slow nonparametric but highly efficient modeling along with Graph-Cut formulation strictly constrains the method to be an offline method. However, a bottleneck in object detection module severely affects real-time capabilities of surveillance systems. The variations of GraphCut formulations are run in GPUs [39] where utilizing such special hardware may not be possible for all surveillance systems.

With this motivation, we propose an alternative last step for getting rid of noise in the images. The work in literature [40] proves that if the image is subject to an average random noise, a median filter can serve as a powerful alternative and give approximately same results with MRF solution. In this work, we replace the last step of the algorithm with an edge preserving weighted median filter algorithm. The filter takes input of  $\tau$  values and sorts pixels of the image. The membership of the pixel is determined using middle



value as the threshold of the weighted filter. Using such median filter removes utilization of  $\delta(x)$  function and selecting an optimal  $\kappa$  function. Results show that optimization using weighted median filter gives very successful results.

## 2.4 Implementation Results

The background of observed scene will change in time so background and foreground models must be updated in time. In the algorithm, a memory of  $M$  frames is kept where last  $p_b$  images are used for updating background model and last  $p_f$  images are used for foreground model. Because foreground model changes faster compared to the background model,  $p_b$  should be larger. Increasing  $p_b$  slows the update process of background model of which stationary objects will take more time to be added to the background model. If kernel density function's bandwidth is enlarged, the algorithm suppresses foreground movement to cope with dynamism. In such a scenario, some parts of the objects can be also considered as member of background model.

In order to implement kernel density function, a 5 dimensional histogram approximation has been selected. As basic filling of histograms was not sufficient for high accuracy, linear binning as described in [41] was utilized. In the work [28], it is indicated that 11 fps is the upper limit where our algorithm exceeds 20 fps. The implementation is single threaded but convenient for multi threaded implementation due to parallel operation. Such an implementation would further boost the real-time performance of the algorithm.

In Figure 2.4, results of implemented algorithm on aquarium scene is shown. The difficulties in this environment were colorful pebbles in the ground of aquarium, reflections of mirrors, abrupt movement of fish during

the algorithm's learning phase (the background model is contaminated with moving objects). The pebbles in the ground of aquarium is a severe problem for algorithms not taking account of neighbourhood information such as GMM and Adaptive Median background subtraction. As shown in Figure 2.4b, Adaptive Median algorithm can suppress pebbles' dynamism however cannot detect all parts of the fish. In Figure 2.4c it is shown that all parts of the fish can be detected but pebbles' dynamism can not be suppressed. Moreover, the shadows of moving objects on the ground of aquarium is another problem to handle. In our approach, all of the fish are successfully detected and dynamism of pebbles' are suppressed (Fig 2.4d). In Figure 2.4, the output of the algorithm on railroad scene, having strong camera jitter, which was used in previous section for comparing real-time background subtraction algorithms has been also shown. It is proved that such nonparametric and real-time approach has noticeably less false detected pixels compared to algorithms in Section 2.1. The output of the algorithm on PETS sequence is shown in Figure 2.4. All objects in the scene are successfully detected.

## 2.5 Discussions

In this chapter several real-time background subtraction algorithms have been compared. Results showed that state of the art GMM based algorithms outperform others. Nevertheless they show great vulnerability in dynamic scenes. To overcome this limitation a nonparametric algorithm which has showed promising results on dynamic scenes has been extended to operate in real-time without losing pixel classification accuracy. The developed algorithm will provide infrastructure for multiple object tracking algorithm mentioned in the next chapter.

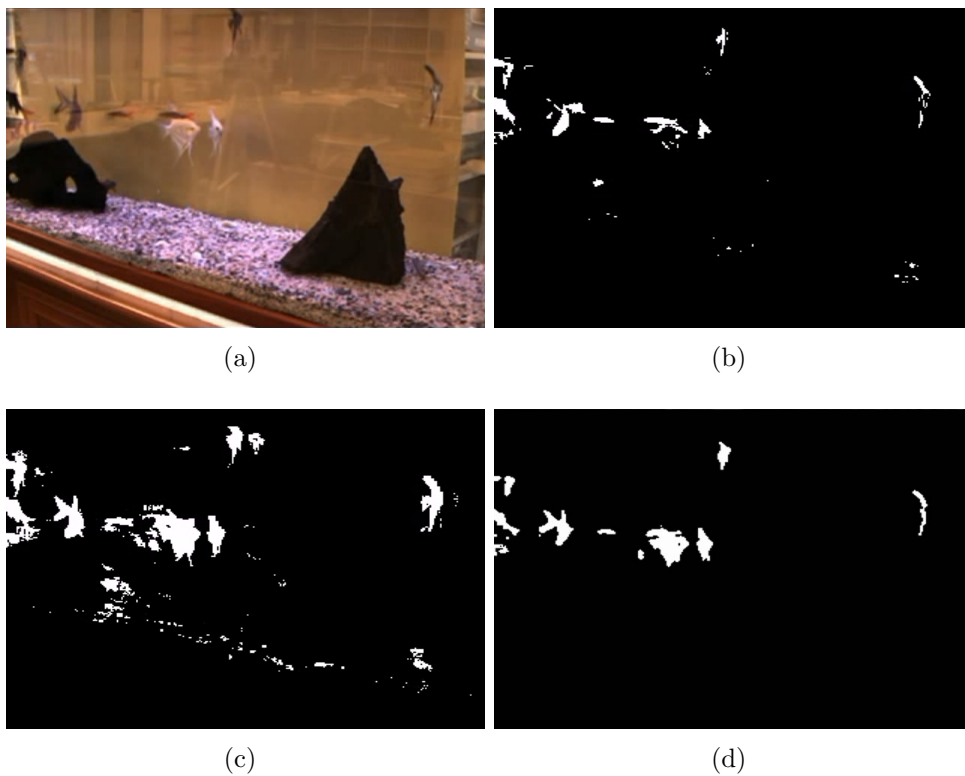
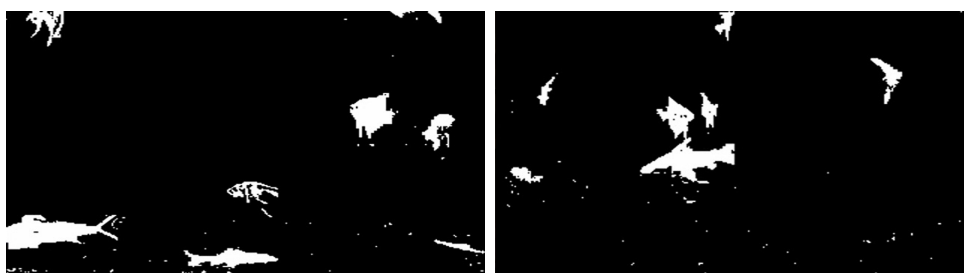


Figure 2.7: a) An image from aquarium dataset b) Adaptive-Median based background subtraction result c) GMM based background subtraction result d) Proposed background subtraction result.



(a) Image



(b) Background Subtraction



(c) 3x3 Median Filter

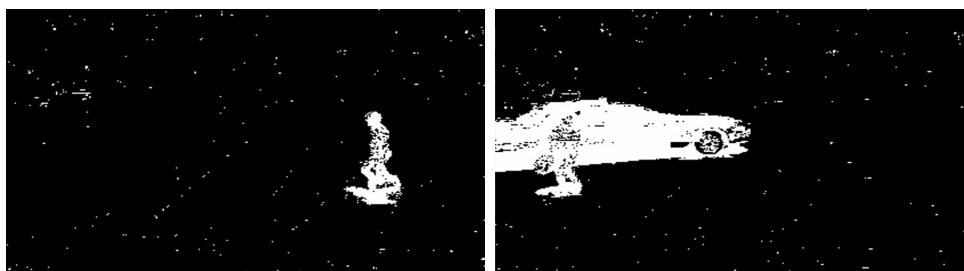


(d) 5x5 Median Filter

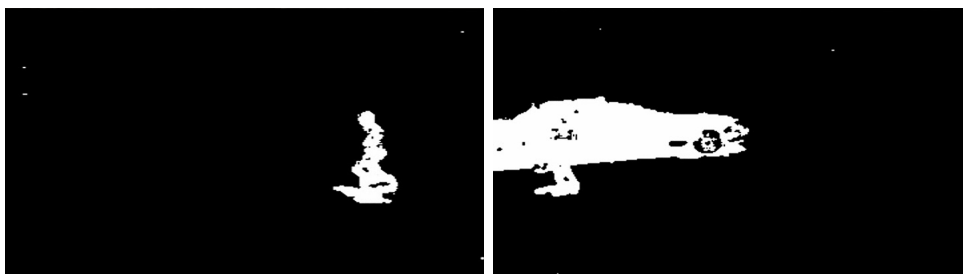
Figure 2.8: a) Two image from aquarium dataset b) Result of Background and Foreground Modeling c) 3x3 Weighted Median Filter as last step d) 5x5 Weighted Median Filter as last step



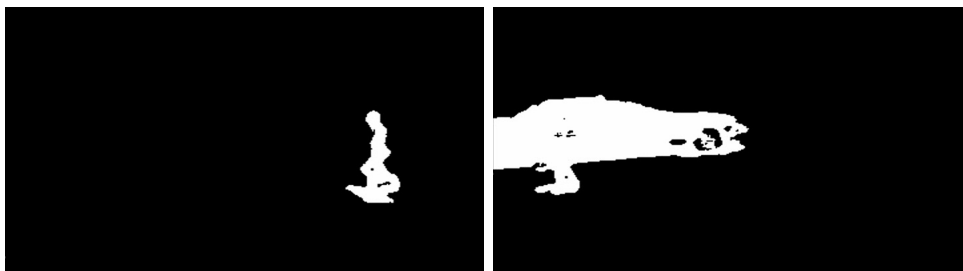
(a) Image



(b) Background Subtraction

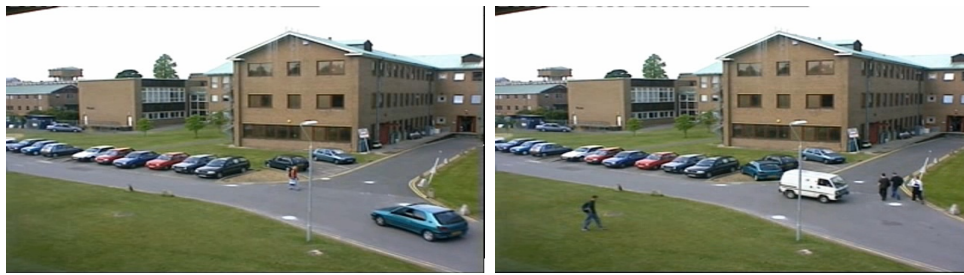


(c) 3x3 Median Filter



(d) 5x5 Median Filter

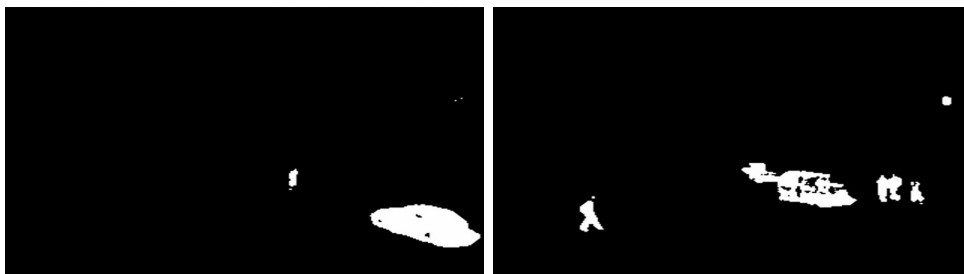
Figure 2.9: a) Two image from railroad dataset b) Result of Background and Foreground Modeling c) 3x3 Weighted Median Filter as last step d) 5x5 Weighted Median Filter as last step



(a) Image



(b) Background Subtraction



(c) 3x3 Median Filter



(d) 5x5 Median Filter

Figure 2.10: a) Two image from PETS dataset b) Result of Background and Foreground Modeling c) 3x3 Weighted Median Filter as last step d) 5x5 Weighted Median Filter as last step

# Chapter III

## 3 Real-Time Multiple Object Tracking Using Virtual Shells

Object tracking has been a focus of research for decades due to its promising potential for real-time applications such as intelligent user interfaces, navigational systems and surveillance. A successful tracking algorithm is expected to be robust against environmental changes. Main difficulties of the single object tracking problem are appearance changes, abrupt motion and object non-rigidity. One can view multiple object tracking as the problem of running multiple tracker instances on each object. However such an approach is likely to fail because of not exploiting the dependence between tracker instances and object models, running each tracker independently, causing tracker to get stuck on one object and lose the others [42]. One needs a more reliable tracker which uses holistic information exploiting the tracking correlation between objects.

Main problem of the multi object tracking systems is object interactions which give rise to occlusions [43, 44]. While simple scenarios usually include interactions of two objects, real world examples may include several objects interacting with each other that give rise to complex events and cause the tracker to fail. This necessitates development of more robust, reliable and easily scalable object tracking formulations. Argyros et. al. [45] propose

online learning of color for tracking skin. Khan et. al. [46] employ particle filtering for tracking of multiple objects. Sullivan et. al. [47] exploit continuity of depth and motion direction for object labeling problem. Yu and Medioni [48] propose a Markov Chain Monte Carlo formulation for multiple object tracking.

Most of the multi object tracking methods, regardless of the number of cameras installed, requires a background subtraction algorithm to detect motion, a prerequisite step for object representation. The occlusion problem is mostly handled in two different ways: merge-split approach and straight-through approach as noted in work of Gabriel et. al. [49]. In the merge-split based methods (McKenna et. al. [50] and Bremond et. al. [51] use appearance cues, Haritaoglu’s W4 system [52] uses appearance and motion cues), separate blobs are updated as long as no occlusion is detected. In the case of occlusion, a joint blob is formed and tracked until the objects are splitted. The detection of beginning and termination of occlusion requires an occlusion and split predicate. Straight-through approaches as in Khan’s method [53] and Haritaoglu’s Hydra system [54] do not handle split and merge cases separately. These approaches continue to track each individual object in the presence of occlusions using various image features. No joint blob, or hypothesis, is formed during the occlusion phase. These methods require an occlusion predicate which acts as a trigger for ongoing occlusion event.

Regardless of the approach that may be taken, a robust and reliable multiple object tracking framework necessitates formal definition of the interaction problem that is boosted by merge and split predicates. Motivated by these observations, we propose a novel method for solving multiple object tracking



problem under severe occlusions. First, occlusion and split predicates are implemented using virtual shells and blobs. Second, a split-merge level analysis is performed using an event resolution step. This step provides information about object interactions with the help of temporal consistence. Third, a straight-through approach is adopted and pixel level analysis is carried out on shells by considering updated events without forming joint objects.

Our tracking approach has three important ingredients: a virtual shell model, an event resolution analysis and a pixel membership evaluation. By a virtual shell it is meant a closed-bounded curve or surface that encloses each object and handles complex interactions between objects that may have arbitrary shapes and motion. An event resolution analysis based on state transitions is performed using geometric relationships between object shells. This resolution step provides a macro level evaluation for multi-object tracking process. Finally, a micro level analysis, namely a pixel membership evaluation is carried out where all interesting pixels of objects are evaluated and assigned to corresponding objects using a probabilistic approach that utilizes virtual shells and event resolutions.

### 3.1 Background Subtraction

In most tracking systems, background subtraction is a preprocessing step for motion detection. In the first step of our algorithm, the background subtraction technique detailed in [37] is applied to the input frame to obtain background and foreground pixels. However, we should remark that one can use any state of the art background subtraction algorithm. For more information about background subtraction techniques for tracking purposes, the reader can check surveys [25, 55].

We use the algorithm in Chapter 2 to obtain pixel labels. As described in the previous chapter, the last step of the detection algorithm is the weighted median filter. Using such a filter guarantees reconstruction of the disconnected parts of an object. We then utilize a connected components labeling algorithm to obtain labeled blobs in the input image. This procedure is then continued with a geometric filtering step which is obligatory to avoid false positive regions. Geometric filtering can be utilized in many different ways. One can create a filter based on angle, shape or size constraints. In a scenario where objects with arbitrary and highly complex shapes that may undergo abrupt motion, a size filter will be adequate to remove false positives of background subtraction process. These enhanced blobs are main regions to create, track and update object hypothesis. However, due to high shape variations in temporal domain, blobs themselves are insufficient for object representation. To overcome this problem, we introduce virtual shells for further processing.

### 3.2 Shell Model

In shell model, each object possesses a unique enclosing shell. Shells provide spatial relaxation, allow prediction of object interactions and possible merge and split events. Object interactions can be interpreted as shell interactions. Object merge events are described as merging of multiple shells. Shell radius defines the interaction range for an object. An object's interaction range is highly correlated to its speed. Thus, rapid objects will have bigger shells whereas slowing down objects' shells will shrink in time as shown in Figure 3.1. Dynamically sized shells facilitate detection of multiple object interactions.

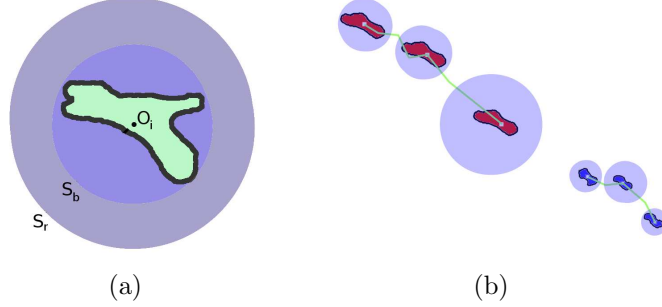


Figure 3.1: a) An object  $O_i$  with its minimum shell  $S_b$  and instantaneous shell  $S_r$  is depicted. b) Shells dynamically grow and shrink with time.

### 3.3 Event Resolution

Object to object relationships in the scene can be in one of three possible states: INDEPENDENT, INTERACTION and JOINT as shown in Fig 3.2. Object state transitions follow a simple but crucial assumption: no two objects can make transition directly from INDEPENDENT to JOINT state or vice versa. An object in JOINT state must first move to the INTERACTION state before becoming INDEPENDENT objects. Thus, the tracking problem can be expressed as keeping track of the states and the positions of each object.

To be more specific, let an object  $O_i$  has an event list denoted by  $L_{i,t}$  at time  $t$ , that keeps record of INTERACTION ( $I_{i,j,t}$ ) and JOINT ( $J_{i,j,t}$ ) relationships with object  $O_j$ . An object having no relationship with other objects has an empty list  $L_{j,t} = \emptyset$ . Proposed shell model provides an infrastructure for the determination of possible state transitions. Let  $S_i$  and  $S_j$  be the shells of the objects  $O_i$  and  $O_j$ , respectively. Also let  $B_i$  and  $B_j$  denote the blobs corresponding to  $O_i$  and  $O_j$ . In what follows, we shall consider several possible scenarios and analyze each in detail.

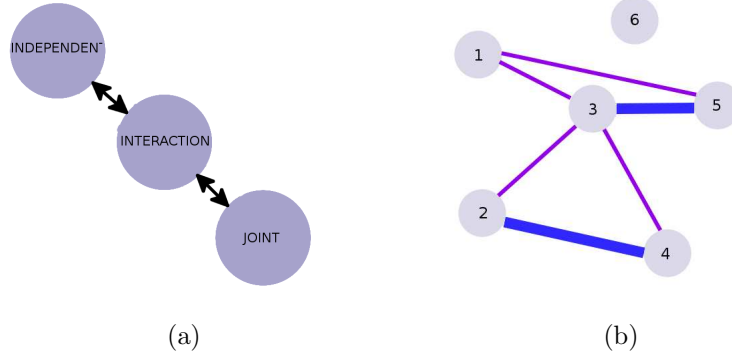


Figure 3.2: a) State transitions of object to object relations. b) State graph where thin edges represent INTERACTION and thick edges represent JOINT relationship between objects.

- Case I.

$$S_i \cap S_j = \emptyset, \quad I_{i,j,t-1}, J_{i,j,t-1} \notin L_{i,t-1} \quad (1)$$

This condition says that if object shells do not intersect and they had no INTERACTION or JOINT relationship at time  $t - 1$ , then each object is INDEPENDENT of each other.

- Case II.

$$\begin{aligned} S_i \cap S_j \neq \emptyset, \quad I_{i,j,t-1}, J_{i,j,t-1} \notin L_{i,t-1} \\ L_{i,t} = L_{i,t-1} \cup \{I_{i,j,t}\} \end{aligned} \quad (2)$$

When two object shells has a non-empty intersection and had no INTERACTION and JOINT relation in their list, one can conclude that this is a start of an INTERACTION event between objects  $O_i$  and  $O_j$ .

Volume of the intersection set accumulates when objects move toward each other. The INTERACTION event must be added to both objects' event list  $L_{i,t}$  and  $L_{j,t}$ .

- Case III.

$$\begin{aligned} B_i &= B_j, \quad S_i \cap S_j \neq \emptyset, \quad I_{i,j,t-1} \in L_{i,t-1} \\ L_{i,t} &= L_{i,t-1} \setminus \{I_{i,j,t-1}\} \cup \{J_{i,j,t}\} \end{aligned} \quad (3)$$

This equation models an incoming JOINT event. If two blobs are identical (single blob), their shells are in interaction, and they were in interaction at time  $t - 1$ , then they have joined into one object. While most of the time object to object relationships can be expressed with either INDEPENDENT or INTERACTION states, JOINT state provides a meaningful and necessary stage when one object fully occludes another one. At the end of occlusion, an occluded object can reappear anywhere but limited to the shell of the occluder object.

- Case IV.

$$\begin{aligned} B_i &\neq B_j, \quad S_i \cap S_j \neq \emptyset, \quad J_{i,j,t-1} \in L_{i,t-1} \\ L_{i,t} &= L_{i,t-1} \setminus \{J_{i,j,t-1}\} \cup \{I_{i,j,t}\} \end{aligned} \quad (4)$$

Two objects in JOINT state can split in time (see Fig 3.3). On split event, their shells will have a non-empty intersection set and their blobs will be spatially distinct. The event list must be updated to hold incom-

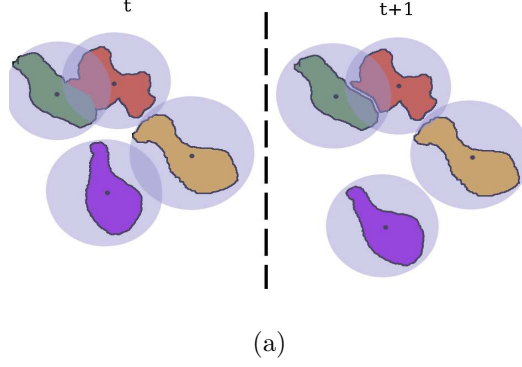


Figure 3.3: Demonstration of Case IV (green and red) and Case V (purple and orange).

ing INTERACTION event and discard previous JOINT event between objects.

- Case V.

$$\begin{aligned}
 S_i \cap S_j &= \emptyset, \quad I_{i,j,t-1} \in L_{i,t-1} \\
 L_{i,t} &= L_{i,t-1} \setminus \{I_{i,j,t-1}\}
 \end{aligned} \tag{5}$$

Two interacting objects end their interaction which is detected by an empty intersection set between shells. In this case both objects will go to a INDEPENDENT state and isolate themselves from each other as shown in Fig 3.3. Previous interaction record must be erased from the event list.

### 3.4 Pixel Membership Evaluation

After performing an event resolution analysis at object level, each object's nearby pixels in a region of interest must be evaluated for pixel membership. This region of interest is selected as object's shell. As in Fig. 3.3, there can be both the object pixels and the interacted objects' pixels inside the shell radius of an object. Each pixel is then evaluated for possible membership with each object  $O_i$  using the following optimization problem.

$$\hat{c}(p_x) = \arg \max_c P(c|p_x), \quad \forall p_x \in S_i, \quad c \in \mathcal{N}(L_{j,t}) \quad (6)$$

where  $\mathcal{N}(\cdot)$  is defined as:

$$\mathcal{N}(L_{j,t}) = \{O_i\} \cup \{O_j | O_j \in L_{j,t}\} \quad (7)$$

Note that the set  $\mathcal{N}$  for an object  $O_i$  is defined as the union of the object and other objects  $O_j$  that are in relations with  $O_i$ . All elements of  $L_{j,t}$  are checked to determine for possible ownership of the pixel  $p_x$ . Then the probability term  $P(c|p_x)$  can be computed by utilizing Bayes theorem; i.e.

$$P(c|p_x) = P(p_x|c).P(c) \quad (8)$$

where the likelihood term  $P(p_x|c)$  is calculated using the color histogram of pixels, and the prior term  $P(c)$  is modeled as a kernel function corresponding to the shell  $S_i$ . In a scenario where frequent occlusions take place and objects undergo abrupt motion, the prior term provides robustness to the pixel evaluation process by imposing the constraint of spatial proximity (to the shell center) on pixels. Pixels far away from the center will have less

attraction to belong to that shell.

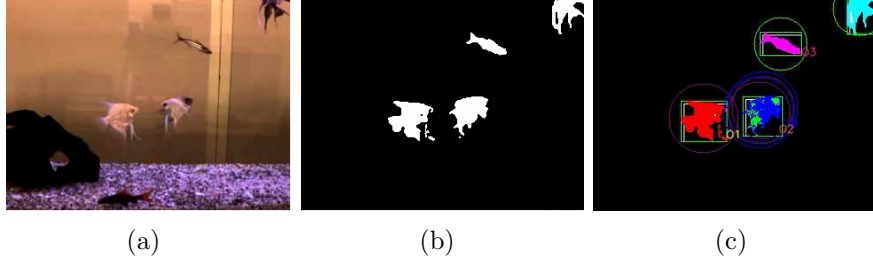


Figure 3.4: a) Aquarium image b) Background subtraction c) Output of the event resolution and pixel membership evaluation where pixels of each object are colored differently for illustration purposes

This kernel function of each object is centered at  $C_{S_i}$ , their shell center. Intuitively this saturation introduces some constraint where an object  $O_j$  can not enter membership voting for a pixel if the pixel is outside of its own shell, even if it is in interaction with  $O_i$ . Selection of the kernel is a question of accuracy and computational constraints. In our implementation, Epanechnikov kernel has been selected as the kernel function. The pixels that are far away from shell center have lower probabilities to belong to that object. The event resolution and pixel membership evaluation steps are presented in pseudo code in **Algorithm 1** and **Algorithm 2**.

As an example, in Fig 3.4, we consider an occlusion scenario between two objects. Purple colored shells indicate interaction where blue color indicate JOINT event between objects. After utilizing pixel evaluation step, green colored pixels indicate pixels of the occluded object while the blue pixels belong to the occluder.



### 3.5 Position Update

After assigning each pixel to the corresponding object, each object’s center is updated using member pixels; i.e.

$$C_{i,t+1} = \alpha E[P_i] + (1 - \alpha) \hat{P}_{C_{i,t+1}|C_{i,t}} \quad (9)$$

where  $P_i$  represents all member pixels of  $O_i$ . The updated position is a linear combination of the current first order moment of  $P_i$ , i.e.  $E[P_i]$ , and the Kalman filter prediction, i.e.  $\hat{P}_{C_{i,t+1}|C_{i,t}}$ . Integration of filtering into the process helps smooth transitions of object centers in temporal domain avoiding instantaneous noisy estimations.

### 3.6 Experiments

We present experiments on PETS, Caviar and aquarium sequences. Our experimental aquarium setup can be seen in Figure 3.5. Aquarium images have been acquired using Imaging Source DFK21BF04H Firewire CCD cameras and resized to 320x240 resolution under RGB color space. The cameras are connected to a desktop computer, which has 2 GB memory and Intel I5 processor, through a 6-Port Firewire hub. We have also developed a graphical user interface (GUI) in QT framework to compare state of the art object detection and tracking algorithms (see Figure 3.5(b-c)). This GUI allows user to select and run detection and tracking algorithms on arbitrary cameras. An ongoing Mean Shift tracking algorithm [11] is depicted in Figure 3.5(b). Our graphical interface provides infrastructure for testing real-time detection and tracking algorithms.

The ground level of aquarium contains shiny colorful pebbles which can

---

**Algorithm 1** Event Resolution

---

```
1: procedure EVENTRESOLVE( $K, L_{k,t-1}$ ) ▷ Updates event list of each object
2:   for  $k = 1 \rightarrow K$  do
3:      $L_{k,t} \leftarrow L_{k,t-1}$ 
4:   end for
5:   for  $i = 1 \rightarrow K$  do
6:     for  $j = i + 1 \rightarrow K - 1$  do
7:       if checkShellJoint( $S_i, S_j$ ) then
8:         if queryInteraction( $L_{i,t-1}, j$ ) then
9:           Add  $J_{i,j,t}$  to event list  $L_{j,t}$ 
10:          Remove  $I_{i,j,t-1}$  from  $L_{j,t}$  and  $L_{j,t}$ 
11:        end if
12:      else
13:        if checkShellInteraction( $S_i, S_j$ ) then
14:          if queryJoint( $L_{i,t-1}, j$ ) then
15:            Remove  $J_{i,j,t-1}$  from  $L_{j,t}$  and  $L_{j,t}$ 
16:            Add  $I_{i,j,t}$  to  $L_{j,t}$  and  $L_{j,t}$ 
17:          else if !queryInteraction( $L_{i,t-1}, j$ ) then
18:            Add  $I_{i,j,t}$  to  $L_{j,t}$  and  $L_{j,t}$ 
19:          end if
20:        else if queryInteraction( $L_{i,t-1}, j$ ) then
21:          Remove  $I_{i,j,t-1}$  from  $L_{j,t}$  and  $L_{j,t}$ 
22:        end if
23:      end if
24:    end for
25:  end for
26:  return  $L_{K,t}$  ▷ Returns event list of each object
27: end procedure
```

---

---

**Algorithm 2** Pixel Membership Evaluation

---

```
1: procedure COMPUTEPIXELCLASSES( $N, M, K$ )  $\triangleright$  Processes NxM image, K objects
2:   for  $x = 1 \rightarrow N$  do
3:     for  $y = 1 \rightarrow M$  do
4:       if  $I(x, y)$  then
5:          $l \leftarrow \text{findOwnerShell}(I(x, y))$ 
6:          $L \leftarrow 0$ 
7:         for all object in  $\mathcal{N}(L_{l,t})$  do
8:            $\theta \leftarrow \text{compute posterior using equation (8)}$ 
9:           store  $\theta$  into list  $L$ 
10:        end for
11:         $b \leftarrow \text{objectIndex}(\max(L))$ 
12:        add pixel  $I(x, y)$  to  $P_b$ 
13:      end if
14:    end for
15:  end for
16:  return  $P_K$   $\triangleright$  P holds each object's pixel list
17: end procedure
```

---

act as false positives to many detection algorithms based on appearance variation. They are suppressed successfully using our background subtraction step as shown in Fig. 3.4b. Moreover, fish make abrupt motion and reflection caused by mirrors of the aquarium is another difficulty challenged in this experimental setup.

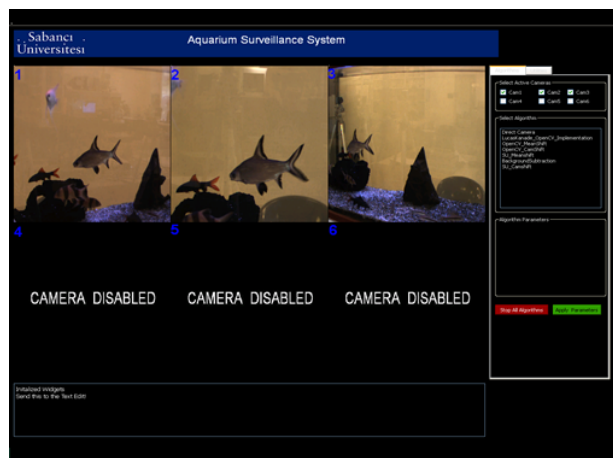
In Fig.3.6 tracker's results on a PETS video is shown. A car and a person is moving towards each other, merges and splits after their interaction ends. In our quantitative analysis, we have used the MOTA metric described in [56] for evaluation of multiple object tracking accuracy. In Fig. 3.7, quantitative results from PETS sequence are shown where three objects (shown by blue, green, red bars) reside occasionally on the scene. We defined the tracking accuracy for each object as the distance of computed object center to the ground truth object center.



(a)



(b)



(c)



Figure 3.5: Our experimental setup: aquarium environment.

A fight scene from CAVIAR database is shown in Fig. 3.8. In this scenario, two people move towards each other and start fighting. During fight as in (Fig. 3.8b-e), severe occlusions are observed in close contact. One man gets down and lies on the floor while other runs away with high speed (Fig. 3.8f-h).

In Fig. 3.9, we show a tracking sequence from our experimental setup; an aquarium containing several fish. The sequence contains rapid motion and self occlusion examples. While in many scenarios split and merge events are solved with the help of linear motion assumptions, this assumption is not valid in this environment due to abrupt and rapid motion. This is observable in Fig. 3.9e-f which shows that small fish in pink bounded box performs a sudden turn. Moreover, self-occlusion and object interactions are very frequent. In Fig. 3.9a fish in blue bounded box severely self-occludes which reduces number of pixels associated with it and suffers from a noticeable shape variation. Interaction of two fish is observable in Fig. 3.9g-h. Fish in blue box appears in front and with maneuver of neighbor fish they change

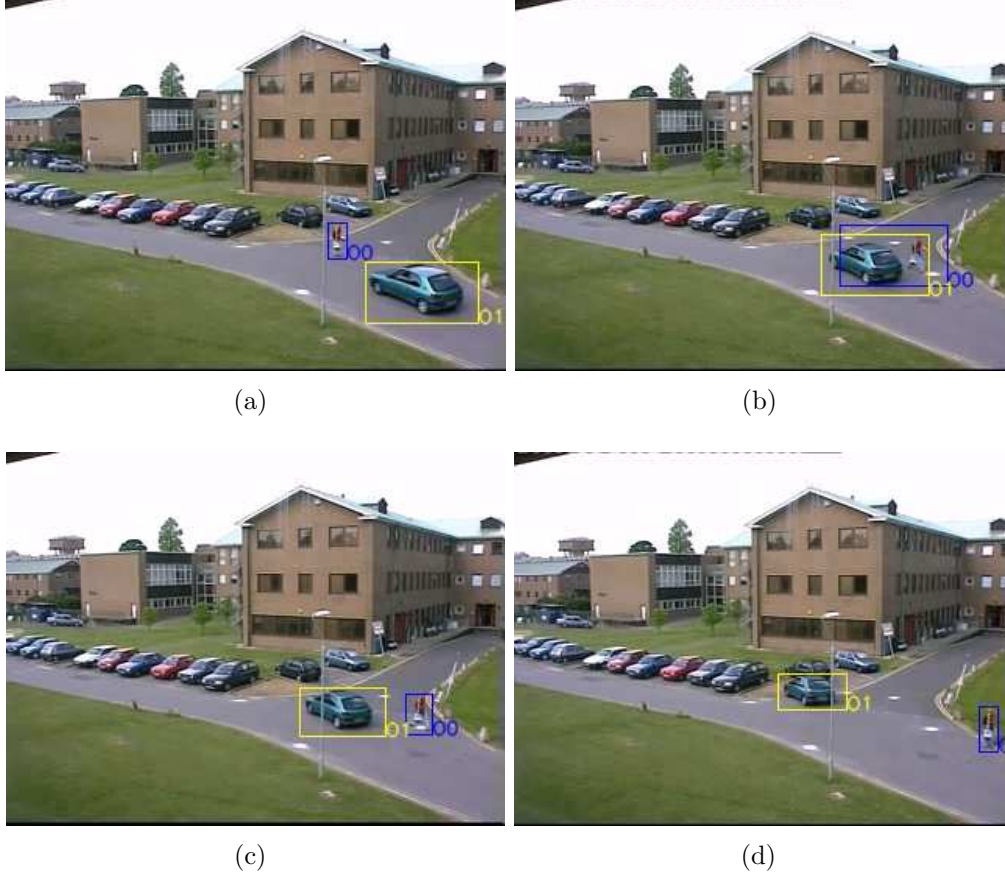


Figure 3.6: Sequence from PETS database.

roles, it appears behind its neighbor in a very short time.

In Fig. 3.10, we show a second sequence from aquarium environment where several objects' interactions occur. This sequence has a total of 350 frames. In this sequence we track six fish. Four of them go into an interaction eventually (Fig. 3.10i-m). Note that objects' interaction complexity is high as many shells collide with each other and pixel classification task extends to several objects. At the end of the interaction, classification succeeds and each fish's location is preserved (Fig. 3.10n-p). This sequence shows that arbitrary

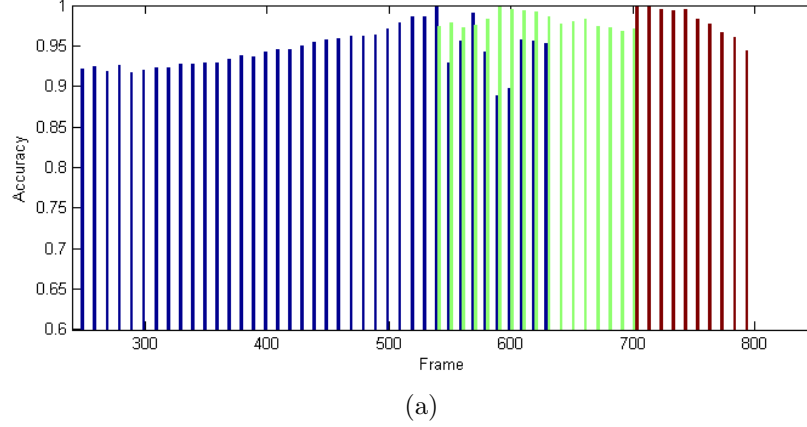


Figure 3.7: Tracking accuracy for PETS Sequence where each color represents a tracked object.

number of objects' interaction can be coped with using this framework.

**Application to UAV Video Data with the help of Video Registration.** Our proposed tracking approach was originally designed for stationary cameras. Nevertheless, with minor modifications it can also be used for moving cameras. As a moving camera observes different regions of the scene, one must first stabilize the motion. Incoming frames can be stabilized in a fast manner using mosaicing technique proposed in Chapter 4. Note that normally, small misregistrations in stabilization process would affect profile of each pixel if we had used GMM based background subtraction as GMM does not take values of neighbour pixels into account for building up intensity profiles. However, using a KDE based background subtraction approach can cope with such small misregistration errors. We showed that KDE approach can cope with camera jitters where each pixel had motion of several pixels.

Most of the color based long term tracking algorithms suffer from illumination changes. In order to cope with illumination changes, the color model





(a)



(b)



(c)



(d)



(e)



(f)



(g)



(h)

Figure 3.8: Sequence from Caviar database



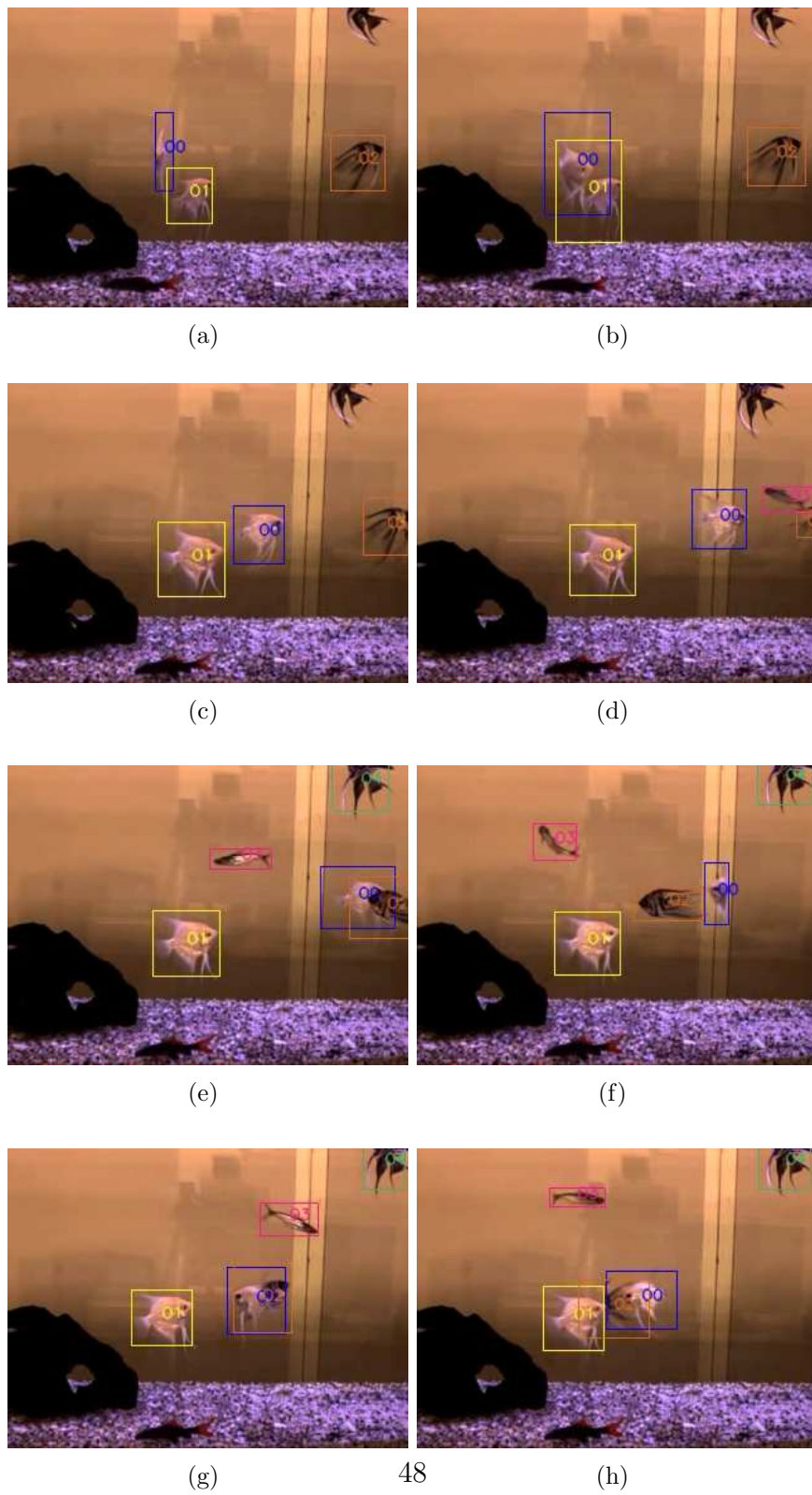


Figure 3.9: A tracking sequence from aquarium where three fish interact

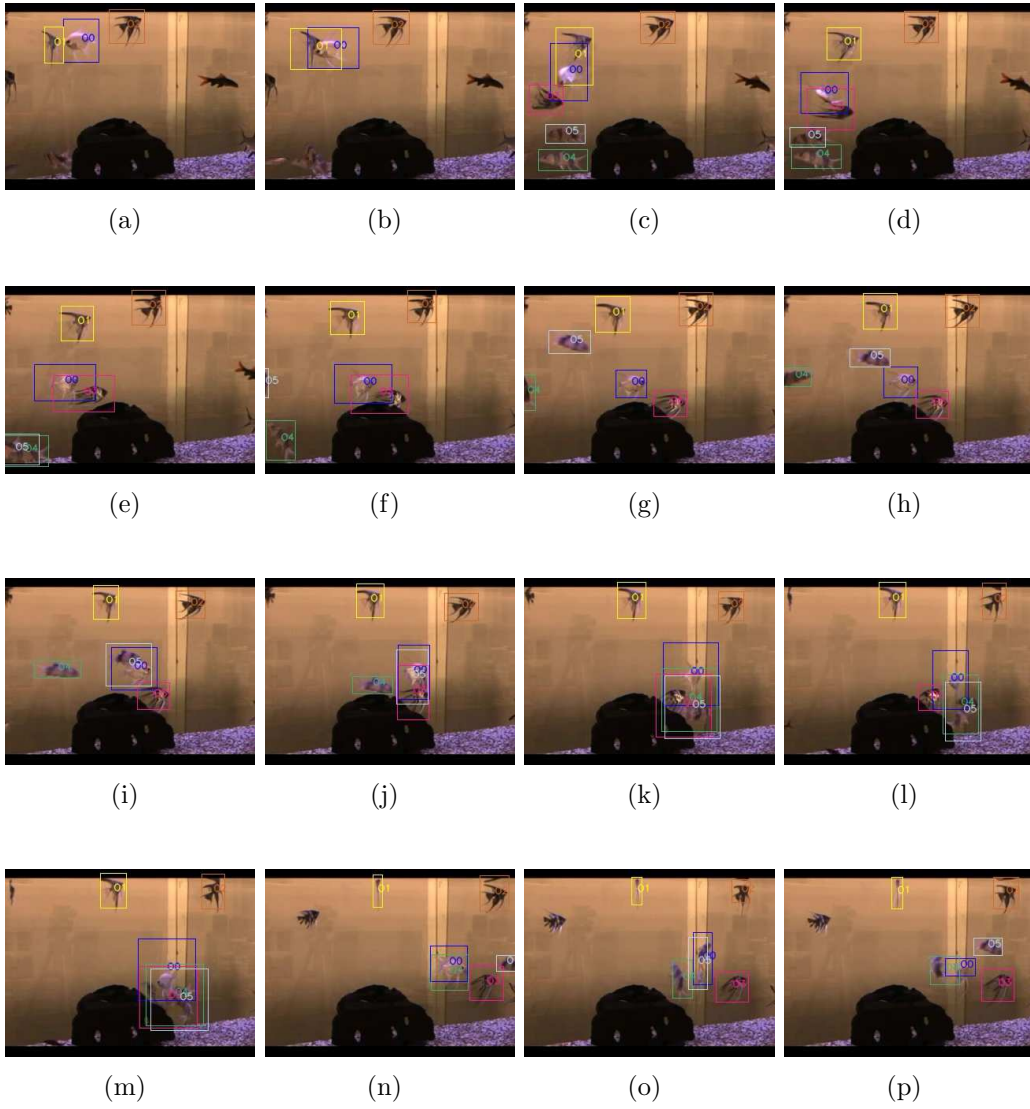


Figure 3.10: A tracking sequence from aquarium where several fish interact

of each object needs to be updated for long tracking sequences. It is more reliable to update objects' color model when objects have an empty event list and no occlusion. In our implementation, we have adopted linear binning for histogram filling technique as mentioned in[57] to improve binning accuracy.

### **3.7 Discussions**

We have now presented a real-time multiple object tracking algorithm. Based on virtual shell modeling, the algorithm uses event resolution analysis and pixel class evaluation to achieve robust tracking. The algorithm has been tested on some well-known databases and also on our challenging aquarium setup where multiple objects interact with each other and create very complicated occlusion scenarios. Algorithm is fast, repeatable and robust. The experimental results are quite promising.

# Chapter IV

## 4 Large Scale Mosaic of UAV Image Sequence

Image mosaicing is the process of stitching many images together in order to create a larger, consistent and seamless composite image. The composite image, having a larger field of view, can provide more information than spatially and temporally distinct separate images. Image mosaics are frequently used in personal, medical and remote sensing applications. Using these algorithms, charming panoramas of natural scenes [58] and office environment can be obtained with inexpensive off-the-shelf cameras. In the context of medical imagery, mosaicing retinal images [59] and tissues [21] produced impressive results. These algorithms are also used for creating large microscopic [60] and fingerprint imagery [61]. For remote sensing purposes, maps of an environment can be created using aerial [62], underwater [63] and satellite images. They are also embedded as image stabilization and video compression routines [64] in video cameras and mobile platforms. First step of virtually all mosaicing method is to perform a highly accurate local alignment between image pairs. In literature, image alignment methods can be classified into two categories as dense or sparse methods. These are also known as direct and feature based approaches [65]. In direct approaches, the whole image data is used instead of sparse features. Within these approaches, transformation parameters and pixel correspondences are estimated simultaneously.

These methods have higher accuracy compared to feature based techniques as all the image information is used. Moreover, they can even exploit uniform regions where no features can be detected. While bringing high accuracy, aligned images must maintain high degree of overlap and initial estimation must be in close proximity to the solution. The pioneering work in this field is done by Lucas and Kanade [66]. A nice overview on historical progress and extensions to this framework can be found in Baker’s work [67]. As exploitation of the whole image data provides rich information, direct approaches are widely used in problems such as mosaicing, tracking and localization. In feature based methods, distinctive image features such as SIFT [23], SURF [68] or affine invariant regions [69] are extracted. After extraction and matching of features, parameter estimation is carried out with the aid of robust estimation techniques. Sparsity of the input data accelerates estimation process and stimulates real-time operation. However it is troublesome to find invariant features in uniform regions in the image and avoid ambiguities caused by repetitive patterns. These problems are studied in the work of Mikolajczyk et al. [70]. Selecting an appropriate transformation model between images is an important step in image registration. A hierarchy of transformations [71] exist under projectivity. Although it is easier to estimate simpler motion models like similarity and affine, having less parameters, these models are only valid under strong camera and scene motion constraints which limit their applicability for a general stitching algorithm. Projective homography is the most general motion model for image stitching. The model is valid under scene planarity or rotational motion constraints [65]. For pure rotational motion, homography is a rotation matrix that has less independent parameters than a full homograph and as a result estimation procedure becomes

more stable [72, 58]. However, this assumption is violated at airborne applications where non-negligible parallax effects are present at low altitudes. Models tackling effects of parallax have been proposed [73, 74, 75]. These plane-parallax models represent the image motion as a mixture of planar and non-planar motion. Applications of plane parallax models are presented for terrain mapping [76] and object detection with UAVs [77]. Apart from the plane-parallax framework, non-negligible parallax effects on mosaic are coped with different methods such as using Graph Cut for depth optimization [78], segmentation of parallax induced regions [79] or selection of appropriate transformations which retain good occlusion handling properties [80].

If the level of deformation between images is noticeably high, a global model may be insufficient for representation of image motion. In such a scenario, a number of local motion models can be estimated [81]. This is especially the case in medical applications where local motion models [82] are popular due to deformative structure of the organs. Although local models can handle high elasticity and non-rigidity in the scene, they are cumbersome due to computational issues making them infeasible for real-time applications.

Ultimate goal of mosaicing methods is to ensure global consistency between all images in the sequence. This is important as barely linking images sequentially in time domain does neglect spatial adjacencies between images. This results in accumulative error propagation through motion model. Spatio-temporal property of image sequence must be fully exploited in order to minimize misregistrations and error accumulations between images while preserving global consistency. This global update step is performed almost in all mosaicing algorithms, especially methods running on large data, either

in a delayed scheme or running concurrently with a local alignment step.

Several different frameworks have been proposed to create attractive mosaics for various scenarios. The analogy of mosaicing to simultaneous localization and mapping problem (SLAM) has been noted by Civera et. al. [83]. Kang formulated the problem in a graph theory framework [84]. Another suggestion is to represent the problem as a reference tracking problem [85] where a reference map reinforces the alignment process in each alignment step.

Out of all mosaicing methods, we primarily focus on mosaicing airborne imagery captured from a UAV. We are interested in the problem of stitching large number of images with small amount of low parallax. There have been offline nonlinear methods for achieving very high accuracy without any timing constraints [86]. The operation can also be done in real-time since frames are taken sequentially. In light of this observation, we aim to stitch large number of images in real-time regardless of the number of images. In what follows, we will review some closely related work to our method. In this Chapter, we propose a mosaicing method for creating seamless mosaics from a set of overlapping separate images acquired from an UAV. The main contribution of our work is to reach a reasonable accuracy on the mosaic without using a computationally expensive framework such as Bundle Adjustment. This is done by exploiting spatial relations between consecutive images and detecting intersections of multiple images using the *Separating Axis Theorem* (SAT). Robust estimation of homographies is done by using MLESAC (Maximum Likelihood RANSAC). The mosaic can be optionally blended to remove misregistrations and photometric defects. The flowchart of our proposed mosaicing algorithm is depicted in Figure 4.1. Proposed

method is expected to run on long image sequences where using other frameworks are inconvenient due to their limited scalability. While there are many studies [87, 88] that boost estimation process with auxiliary data, we avoid such an approach and do not perform any correction with non-visual on-board sensors. This will increase usability when sensorial data is inaccurate or unavailable. Although our method is validated on a set of overlapping images (the ratio varies between %60-%90), same algorithm can be applied to a video sequence where frame to frame overlap is very high with minor additions such as keyframe generation.

## 4.1 Pairwise and Warped Alignment

Image mosaicing involves transforming images captured from different camera poses as if they are taken from a single camera and registering them on a single image plane which is the reference frame. Although there is no quantitative evaluation in the literature on how selection of reference frame effects quality of large scale mosaics, there has been some work to select reference frame dynamically [59]. In this work, we select the first image as reference image for simplicity. The simplest way to register sequential images acquired from a UAV is to perform pairwise matching between successive images and using the homographies of these image pairs to align them. Pairwise matching of the successive images has merit since these kinds of image pairs are supposed to have large number of feature matches. To create the mosaic, we need to link every image to the reference image. Given  $n$  images of a planar scene  $I_0, I_1, I_2, \dots, I_{n-1}$  and assuming  $n - 1$  pairwise homography matrices  $H_{01}, H_{12}, H_{23}, \dots, H_{(n-1)n}$  related to images are known where  $H_{ij}$  is the transformation aligning  $I_j$  to  $I_i$ , we can calculate the homography between



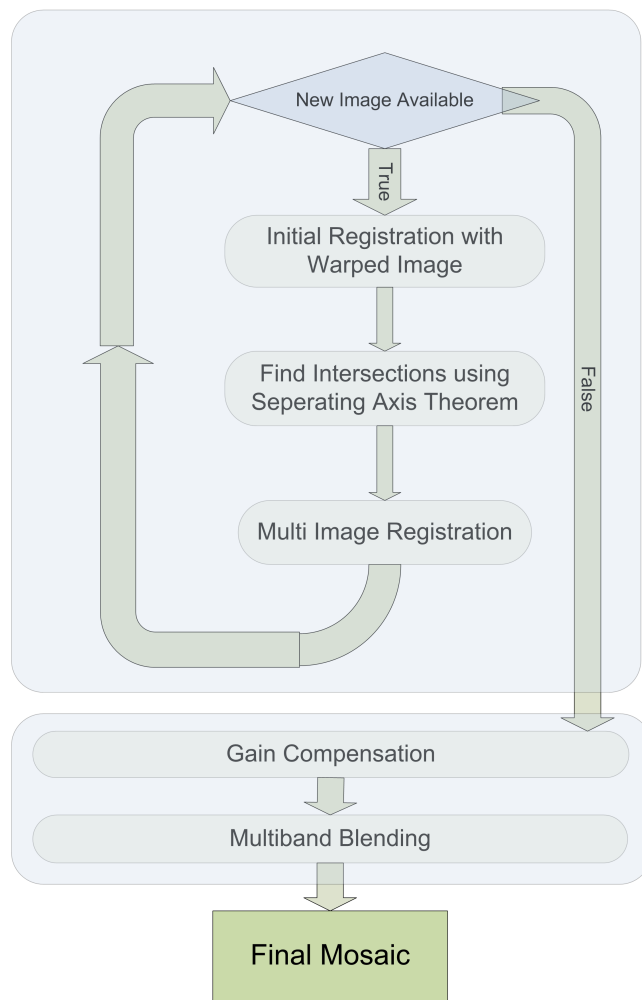


Figure 4.1: Flowchart of our mosaicing approach.

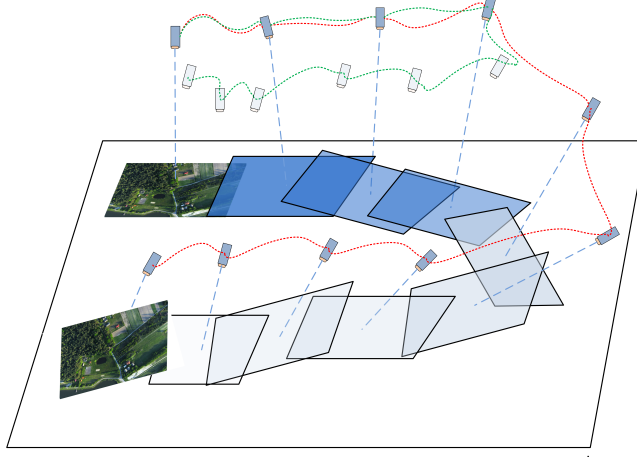


Figure 4.2: Drift caused by estimation errors. UAV returns to same area and snaps same image from initial position. True and estimated trajectories are shown with green and red dashed curves respectively.

any two images,  $I_m$  and  $I_k$ , as:

$$H_{km} = \prod_{i=k}^{m-1} H_{i(i+1)} \quad (1)$$

which is a recursive multiplication of all previous pairwise homographies. To create the mosaic, all the images can be registered to the reference image using homographies calculated from this equation. Although this approach seems to be straightforward, alignment errors tend to accumulate very fast. This is because of the multiplicative nature of the homography calculation and the metric used to estimate the pairwise homographies. Since we use Direct Linear Transformation (DLT) algorithm to estimate the pairwise homographies, our estimation has the minimized algebraic error for the feature matches. Cost function can be given as:

$$\|x_i - H_{ij}x_j\|^2 \quad (2)$$

In this cost function, error is defined on the image  $I_i$ . Since a projective transformation is to be applied on the image to register it on the mosaic, estimated pairwise homography between image  $I_i$  and  $I_j$  doesn't have the minimum error property anymore because norm of the residual vectors are also changed due to projective transformation. A better approach is to estimate homographies of the new images between the image and the mosaic directly. In this case, homography estimation is carried out between the mosaic and the new image. However, as the mosaic grows with every new image, feature extraction and matching process needs more computational power. Furthermore, image features are distorted because of the alignment which can lead to less number of feature matches between images. As a result this approach has its own disadvantages. As a solution to this problem, two approaches can be fused to preserve the advantages of both methods. In what follows, feature matches are found in a pairwise fashion but estimation is performed after transforming features to mosaic frame. In other words, the features of new image  $I_i$  is extracted and feature matches are found with  $I_{i-1}$ . Then features of  $I_{i-1}$  image is transformed to the mosaic frame and homography estimation  $H_{0i}$  is carried out using the cost function

$$\|H_{0,i-1}x_{i-1} - H_{0i}x_i\|^2 \quad (3)$$

where  $x_i$  and  $x_{i-1}$  are feature matches between  $I_i$  and  $I_{i-1}$ . Using this approach, we preserve the matching quality of the images and save computational power using pairwise matching. The approach prevents error accumulation by using image-to-mosaic match. Results for these two approaches can be seen in Figure 4.3. We see great reduction in error propagation and the

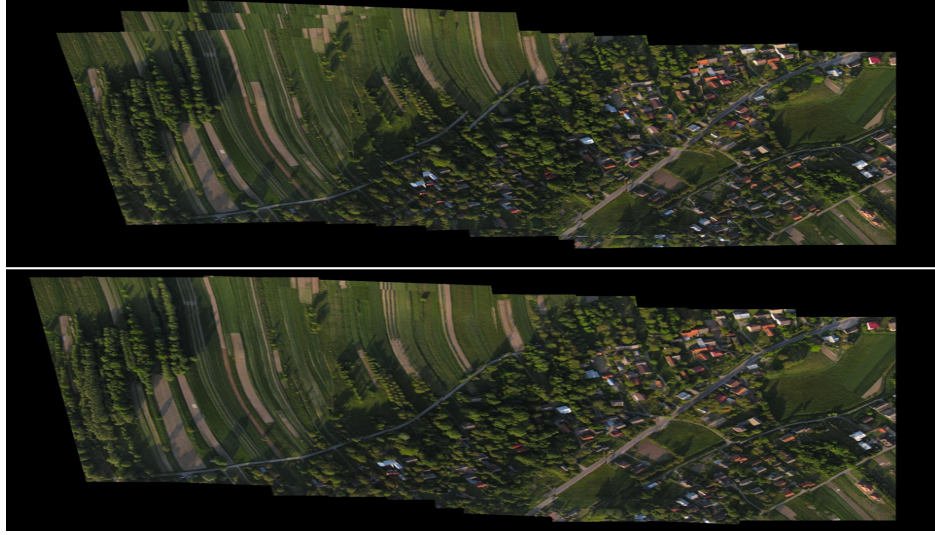


Figure 4.3: Comparison of pairwise stitching using Equation 1 and stitching using cost function 3 (bottom). Note that misregistrations caused by error in the leftmost frames are prevented in the second case.

constructed mosaic is more consistent.

## 4.2 Robust Estimation

Normally, estimation of homography parameters from feature correspondences are carried out with RANSAC [89] due to its outlier elimination capabilities. As RANSAC penalizes and costs each feature match equally regardless of their match quality under estimated model, valuable information content is discarded. Also, there is no modeling of noise in RANSAC formulation which is essential for cluttered scenes containing many boundary level features. In other words, features that reside on facades or roofs of buildings are likely to have high feature localization errors. In what follows we improve sequential homography estimation by using MLESAC (Maximum Likelihood Estimation RANSAC) [90]. Such a statistical model for localiza-

tion and matching errors is expected to bring better accuracy, especially in scenes having more clutter. When one has no prior information about underlying process, maximum likelihood estimation can be used. In feature matching process we do not know underlying noise model, so without loss of generality it is appropriate to model matching of inlier features' noise as Gaussian. Let  $r_{i,j}$  be residual of  $i$ th feature match in  $j$ th image,

$$r_{i,j} = (\underline{x}_i^j - x_i^j)^2 + (\underline{y}_i^j - y_i^j)^2 \quad (4)$$

where underlined variables represent noise free coordinates. For a given motion  $M$ , the probability density function of a noise perturbed data containing residuals can be given as

$$Pr(D|M) = \prod_{i=1..n} \left( \frac{1}{\sqrt{2\pi}\sigma} \right)^n e^{-(r_{i,1}+r_{i,2})/(2\sigma^2)} \quad (5)$$

where  $n$  is the number of correspondences and  $M$  is the underlying motion model between correspondences and  $D$  is the set of matches. The negative log likelihood of any correspondences  $x_i^{1,2}$  where  $i = 1..n$  can be given as:

$$- \sum_{i=1..n} \log(Pr(x_i^{1,2}|M, \sigma)) = \sum_{i=1..n} \sum_{j=1,2} r_{i,j} \quad (6)$$

as  $\sigma$  is constant for all matches, constant term in Eqn. (5) can be dropped. Minimizing this log likelihood corresponds to making *Maximum Likelihood Estimation* of the data. The second step is to also perform noise modeling for outliers. As we have no prior information on outliers, and outliers can occur on each feature, they are modeled as uniform distribution. If we denote total error of feature match  $i$  as  $e_i = \sum_{j=1,2} r_{i,j}$  then resulting mixture model

containing noise model for inliers and outliers can be given as:

$$Pr(e) = \left( \gamma \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{e}{2\sigma^2}\right) + (1 - \gamma) \frac{1}{v} \right) \quad (7)$$

where  $\gamma$  is mixture parameter,  $\sigma$  is standard deviation of error,  $v$  is the search window of outliers. The parameters of  $\gamma$  and  $v$  can be estimated using an Expectation Maximization algorithm [32] which is known to be very fast. Typically, in 2 or 3 iterations EM algorithm converges. In fact, Tordoff [91] empirically showed that estimation of these parameters slightly affect the model estimation results meaning that  $\gamma$  can be set to 0.5 for further speed. Empirically, we have observed that EM algorithm typically estimates  $v$  as  $\sim 15$  pixels. The final log likelihood of the model can be given by

$$-L = - \sum_i \log \left( \gamma \left( \frac{1}{\sqrt{2\pi\sigma^2}} \right)^n \exp \left( -\frac{e_i}{(2\sigma^2)} + (1 - \gamma) \frac{1}{v} \right) \right) \quad (8)$$

Such a formulation penalizes inliers depending on how well they fit to data under estimated motion. This is important when working with quasi-planar scenes where high parallax features will be penalized accordingly. We use normalized DLT under MLESAC framework to better approximate geometric error [71].

### 4.3 Multi-Image Alignment

Although optimization discussed in Section 4.1 implies better results, registering  $I_i$  using information content only from previous image is not sufficient and error-prone, especially when camera is not calibrated and has a significant amount of distortion. It is possible that  $I_i$  may also has feature matches

with  $I_j$  where  $0 < j < i$ . The mosaic consistency can be improved by using information content from these neighboring images. However, number of images to search for matches increase linearly in large scale mosaics. Since searching matches between images is a computationally expensive procedure, which is prohibitive for real-time purposes, we need to envelope search space for possible match candidates. As we know the approximate warped coordinates of  $I_i$ , it is reasonable to search common features from  $I_j$  which are in close proximity of  $I_i$  since spatially distant images are unlikely to have matches with  $I_i$ . Let  $I_i$  be initially registered to the mosaic using  $\hat{H}_{0i}$ , as in Section 4.1. Then, estimation is robustified by using matches from neighbour images  $I_j$ . As neighbour images intersect with current image, an efficient way to detect these intersections is necessary. For this purpose, we utilize Separating Axis Theorem (SAT), a popular tool in computer graphics which is used for collision detection purposes [92]. The theorem simply states a test for nonintersection of two convex objects. If there exists a line for which the intervals of projection of the two objects onto that line do not intersect, then the objects do not intersect. Such a line is called a separating line or, a separating axis. The translation of a separating line is also a separating line, so it is sufficient to consider lines that contain the origin. Given a line containing the origin and with unit-length direction  $\vec{d}$ , the projection of a convex set  $C$  onto the line is the interval

$$Int = [\lambda_{min}(\vec{d}), \lambda_{max}(\vec{d})] = [\min\{\vec{d} \cdot \vec{X} : \vec{X} \in C\}, \max\{\vec{d} \cdot \vec{X} : \vec{X} \in C\}] \quad (9)$$

where possibly  $\lambda_{min}(\vec{d}) = -\infty$  or  $\lambda_{max}(\vec{d}) = +\infty$ ; these cases arise when the convex set is unbounded. Two convex sets  $C_0$  and  $C_1$  are separated if there

exists a direction  $\vec{d}$  such that the projection intervals  $Int_0$  and  $Int_1$  do not intersect. Specifically they do not intersect when

$$\lambda_{min}^0(\vec{d}) > \lambda_{max}^1(\vec{d}) \text{ or } \lambda_{max}^0(\vec{d}) < \lambda_{min}^1(\vec{d}) \quad (10)$$

where the superscript corresponds to the index of convex set. For 2D mosaics, estimated transformations preserve convexity of the images. For these reasons, using this theorem is legitimate for collision detection of warped images. An illustration of the theorem is depicted in Fig. 4.4. By utilizing Separating Axis Theorem, we obtain  $m$  images intersecting with current image,  $I_i$ . In what follows, correspondence list of MLESAC is concatenated using  $I_j$ 's mutual correspondences with the current  $I_i$ . After these steps we have a longer and more robust correspondence list which will further tune  $\tilde{H}_{0i}$ . This step is similar to making a measurement about local portion of the mosaic. The output of the multi image matching step is the final homography  $H_{0i}$  which warps  $I_i$  to the mosaic. We observe that consistency greatly improves by using all neighbours. Moreover, some of the neighboring images can provide richer context (they may have less mismatches and less distorted), feature matches of these images with current image must be weighted higher in the estimation. In the next section we show how to select these weights effectively.

## 4.4 Offline Enhancements

After obtaining all images, we can postprocess the mosaic by performing gain compensation and multiband blending operations. These two procedures remove visually distinct seams and increase visual quality of the mosaic. Nev-



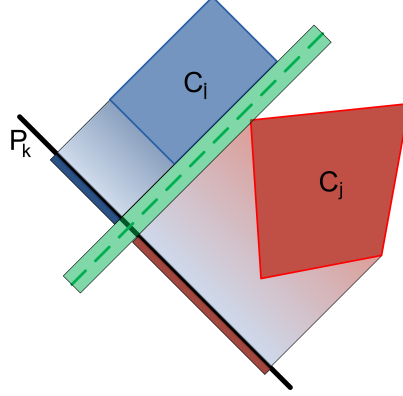


Figure 4.4: An illustration of SAT. For a selected projection axis  $P_k$ , projected convex sets did not intersect. Dashed green line is an example separator for this projection axis.

ertheless they can not be performed in realtime because of their noncausality. We need to acquire all images, and run these two steps in offline fashion. After obtaining homographies that warp each image to the mosaic reference, we can observe illuminance differences between warped images. While UAV flies over an area significant illumination differences can be observed. Main reason of this phenomenon is camera gain variance. Also, variation in flight altitude effects the illumination conditions. In order to obtain visually attractive mosaics, these illumination differences must be corrected. Intuitively, the illumination corrected images must have minimum intensity difference for all pixels in their overlapping regions. This can be mathematically denoted as

$$e = \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \sum_{p_k \in \{I_i^w \cap I_j^w\}} (g_i I_i^w(p_k) - g_j I_j^w(p_k))^2 \quad (11)$$

where camera gains are denoted by  $g_i, g_j$  and  $p_k$  is an arbitrary pixel in overlapping region of both images. Because intensity values of overlapping pixels may be noisy and performing optimization over all pixels is harder,

intensity of the overlapping region can be approximated by the mean value:

$$I_{ij} = \frac{\sum_{p_k \in \{I_i^w \cap I_j^w\}} I_i(p_k)}{\sum_{p_k \in \{I_i^w \cap I_j^w\}} 1} \quad (12)$$

Since solution of Eqn. 11 has a trivial solution that is  $g = 0$ , we have to avoid it by introducing a prior term to the system. In what follows, the final system of gain compensation step can be given as

$$e = \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n N_{ij} \left( \frac{(g_i I_{ij} - g_j I_{ji})^2}{\sigma_N^2} + \frac{(1 - g_i)^2}{\sigma_g^2} \right) \quad (13)$$

where  $N_{ij}$  is the number of pixels in corresponding overlapping region. The parameters  $\sigma_N^2$  and  $\sigma_g^2$  are variances of normalized intensity errors and gain respectively. In our implementation, we selected  $\sigma_N^2 = 10$  and  $\sigma_g^2 = 10^{-4}$ . The system is quadratic in gain parameters  $g$  and can be solved in closed form by setting its derivate to zero. **Multi Band Blending.** Although gain compensation step eliminates severe illumination differences, we still observe seams caused by slight illumination differences and vignetting effects due to unmodeled effects such as parallax and radial distortion. Moreover small misregistrations occur in the mosaic and a blending technique is required to act as a make up. The input to the blending algorithm is  $N$  images  $I^i$ , expressed in planar coordinate frame. Information from multiple frames can be fused using a weight function. The weight function can be denoted as  $W(x, y) = w(x)w(y)$  where  $w(x)$  and  $w(y)$  varies linearly between 0 at the edges and 1 at the image centre. This weight functions are also resampled by warping them on the mosaic frame which is denoted as  $W^i(x, y)$ . Now the problem is to fuse all images given corresponding weight functions. A

straightforward idea is to combine all these images with resampled weights:

$$M^{linear}(x, y) = \frac{\sum_{i=1}^N I^i(x, y) W^i(x, y)}{\sum_{i=1}^N W^i(x, y)} \quad (14)$$

where  $M^{linear}(x, y)$  is the mosaic image formed by combining all input images using linear blending technique. Although this approach is simple, it can fail to mask small misregistration errors. In what follows, regions having small misregistrations will have a blur effect. This can be prevented using a multi-band blending technique such as [93]. The blending weights for each image are initialized by finding the set of points for which image  $i$  is most responsible.

$$W_{max}^i(x, y) = \begin{cases} 1 & \text{if } W^i(x, y) = \operatorname{argmax}_j W^j(x, y) \\ 0 & \text{otherwise} \end{cases} \quad (15)$$

In other words, each image will only claim weight on pixels that it has maximum weight out of all overlapping competitors.  $W_{max}^i$  will be 1 for those pixels, and for all other pixels the weights will be set to 0 in which another image has higher weight. The weight maps are further blurred for each band to form weights in each band. A high pass version of the image can be formed by

$$B_{\sigma}^i(x, y) = I^i(x, y) - I_{\sigma}^i(x, y) \quad (16)$$

$$I_{\sigma}^i(x, y) = I^i(x, y) * g_{\sigma}(x, y) \quad (17)$$

where  $g_{\sigma}(x, y)$  is the Gaussian filter with standard deviation  $\sigma$ . The image  $I^i(x, y)$  is convolved with a Gaussian and subtracted from itself to form high

pass version of the image, which preserves less details.  $B_\sigma^i(x, y)$  represents spatial frequencies in the range of wavelengths  $\lambda \in [0, \sigma]$ . For this band, the images must be convolved with corresponding max weight functions

$$W_\sigma^i(x, y) = W_{max}^i(x, y) * g_\sigma(x, y) \quad (18)$$

where  $W_\sigma^i(x, y)$  represents blending weight for the wavelength  $\lambda \in [0, \sigma]$ . In what follows, each subsequent band  $k \geq 1$  is blended using previous lower frequency band images and weights

$$B_{(k+1)\sigma}^i = I_{k\sigma}^i - I_{(k+1)\sigma}^i \quad (19)$$

$$I_{(k+1)\sigma}^i = I_{k\sigma}^i * g_{\sigma'} \quad (20)$$

$$W_{(k+1)\sigma}^i = W_{k\sigma}^i * g_{\sigma'} \quad (21)$$

where the Gaussian standard deviation of next band is set as  $\sigma' = \sqrt{2k+1}\sigma$ . As a result, subsequent bands have the same range of wavelengths. The final composite for each band is formed as:

$$I_{k\sigma}^{multi}(x, y) = \frac{\sum_{i=1}^N B_{k\sigma}^i(x, y) W_{k\sigma}^i(x, y)}{\sum_{i=1}^N W_{k\sigma}^i(x, y)} \quad (22)$$

The multiband mosaic is obtained by summing images of all subsequent bands. This multiband blending approach allows high frequency bands to be blended over short ranges. Low frequency bands are blended over larger ranges. The final mosaic after blending step is shown in Figure 4.5.

## 4.5 Experimental Results

In our implementation, we operate on a pano of 4500x4500. For faster computation, input images are resized to resolution of 320x240 where more than 1000 SIFT features can be found on average. We use publicly available aerial image dataset [94]. The dataset contains approximately 600 sequential images having resolution of 3648x2736 shot from a Pteryx UAV platform using Canon PowerShot S45. Frames are shot with less than  $5^\circ$  roll and pitch angles. Some of the frames contain geotagging information provided by onboard GPS and IMU. Overlap ratio varies through the image sequence. We observe slight variations on camera gain across the sequence. The calibration data is not provided for the dataset. The output of our method on Czyste sequence is shown in Figure 4.5. Another dataset we use is UAVPeople dataset [95] that is also publicly available. We show results on two aerial image sequence, MunichQuarry and Eastern Island data. In Figure 4.6 the output of our approach is shown for Munich Quarry sequence. The final mosaic of Eastern Island sequence is depicted in Figure 4.7. The proposed approach shares limitations with other feature based methods, namely it is hard to stitch when no features are found.

## 4.6 Discussions

In this chapter, we aimed to create consistent image mosaics from a set of ordered images with real time capabilities. First, instead of using pair wise alignment to warp a new image to the mosaic which is shown to create big inconsistencies after warping a small number of images, alignment of the every new image is performed using the warped images. We observed that this

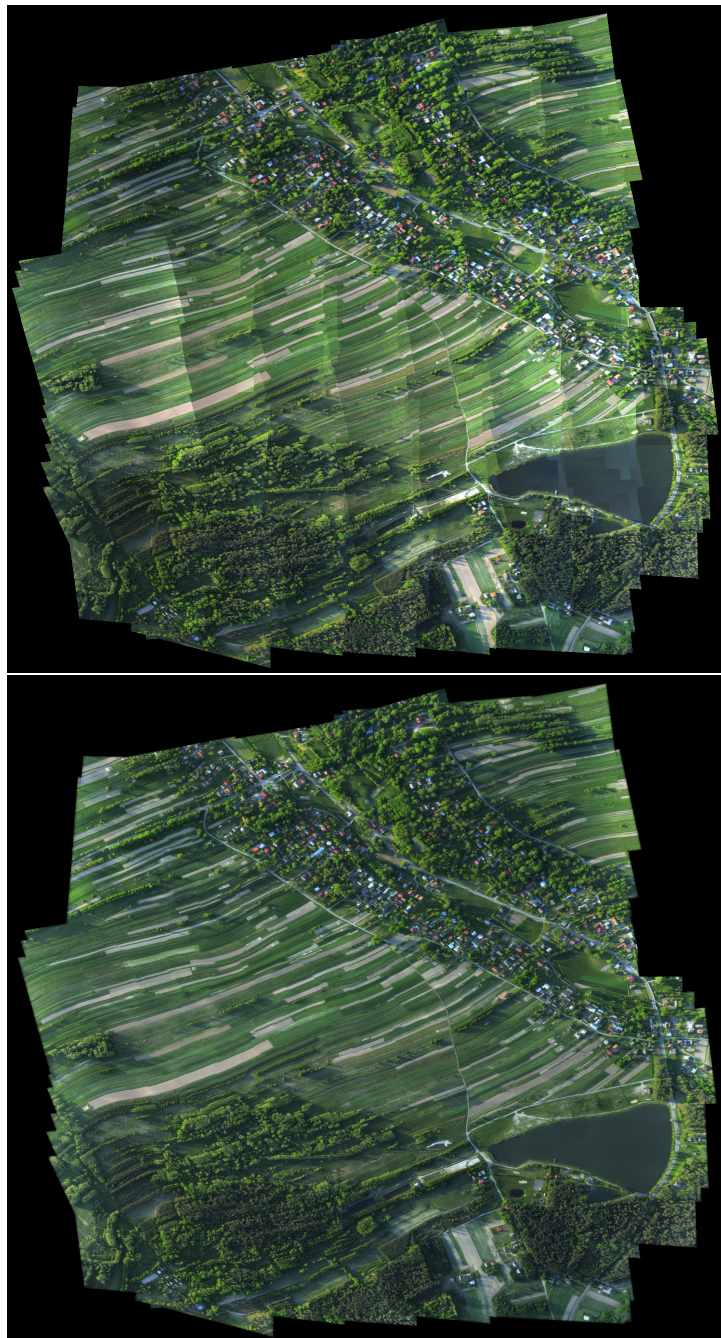


Figure 4.5: Czyste sequence. Images represent before and after blending respectively.



Figure 4.6: Final aerial mosaic constructed from Munich Quarry sequence

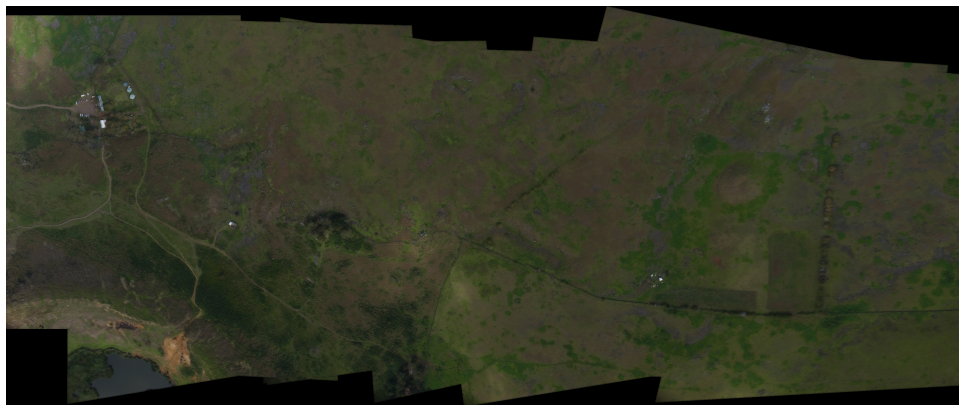


Figure 4.7: Final aerial mosaic constructed from Eastern Island sequence

approach introduces a substantial amount of improvement to the mosaic. As a second step, during the homography estimation procedure, instead of using RANSAC, a sequential homography estimation procedure which is known as MLESAC is used. Additionally, using all the previous images which have common texture with the new image is taken into account during warping process to add the new images to the mosaic. Since it becomes computationally expensive to check for common features from all the previous images, a two phase elimination procedure is performed to get the possible candidates. First step is to index all the warped images in a 2D histogram using their center coordinates. For a new image, only images from the spatially neighbor bins are considered as candidates. To further reduce the number of these images, 'Separation Axis Theorem' is used. In the end, unnecessary matching procedures are avoided and the relevant images with the new image are found in an efficient way. As a result, A mosaic of a sequence with 500 images can be created with acceptable level of inconsistencies for a real-time mosaicing algorithm. After performing gain compensation and multiband blending procedures as offline steps, better results are obtained, which are nearly comparable to the results of offline mosaicing algorithms.



# Chapter V

## 5 Conclusions and Future Work

We have now presented object detection, tracking and image mosaicing algorithms that can be used in many different applications including visual surveillance systems. Proposed detection algorithm shows promising results in dynamic scenes such as an aquarium in which its counterparts have failed to eliminate misdetections caused by dynamic background. Moreover, the proposed algorithm is used as a framework for our multi object tracking algorithm. We have developed a multi object tracking algorithm based on virtual shell modeling to formulate data association problem of several objects. Our proposed tracking algorithm is tested on several publicly available datasets and our aquarium setup where multiple object interactions with severe occlusions take place. The results show that proposed algorithm can cope with such occlusions quite successfully. Lastly, we have presented an image mosaicing algorithm which can create mosaic of an input image sequence efficiently without utilization of non-linear minimization techniques. Experimental results show the success of the proposed mosaicing technique on several UAV image sequences.

Both detection and tracking algorithms were originally designed for stationary cameras. However, with proposed image mosaicing algorithm, these two algorithms can also be extended to work with moving cameras. All of

the algorithms proposed in this thesis runs in real-time.

As a future work, one can integrate several robust cues along with color for an improved pixel discrimination, devise better handling mechanisms for disputed pixels, and develop soft assignment rules for pixels to increase accuracy of the tracking algorithm. Detection, tracking and mosaicing algorithms can also be combined to build a fast and robust visual surveillance system.

## References

- [1] Intelligent Video Surveillance, VCA & Video Analytics: Technologies & Global Market. <http://www.reportlinker.com/>.
- [2] Weiming Hu, Tieniu Tan, Liang Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 34(3):334–352, 2004.
- [3] Christian Micheloni, Marco Lestuzzi, and Gian Luca Foresti. Adaptive video communication for an intelligent distributed system: Tuning sensors parameters for surveillance purposes. *Mach. Vision Appl.*, 19(5-6):359–373, September 2008.
- [4] G. Galati, M. Ferri, P. Mariano, and F. Marti. Advanced integrated architecture for airport ground movements surveillance. In *Radar Conference, 1995., Record of the IEEE 1995 International*, pages 282–287, 1995.
- [5] J.E. Boyd, J. Meloche, and Y. Vardi. Statistical tracking in video traffic surveillance. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 163–168 vol.1, 1999.
- [6] Omar Javed and Mubarak Shah. *Automated Multi-Camera Surveillance: Algorithms and Practice*. Springer Publishing Company, Incorporated, 1 edition, 2008.

- [7] Jian Yao and J. Odobez. Multi-layer background subtraction based on color and texture. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, 2007.
- [8] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511–I–518 vol.1, 2001.
- [9] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4), December 2006.
- [10] I.F. Sbalzarini and P. Koumoutsakos. Feature point tracking and trajectory analysis for video imaging in cell biology. *Journal of Structural Biology*, 151(2):182 – 195, 2005.
- [11] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(5):603–619, May 2002.
- [12] Michael Isard and Andrew Blake. Conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.
- [13] Richard Szeliski and Heung-Yeung Shum. Creating full view panoramic image mosaics and environment maps. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques, SIGGRAPH '97*, pages 251–258, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

- [14] Carlo S. Regazzoni, Gianni Vernazza, and Gianni Fabri, editors. *Advanced Video-Based Surveillance Systems*. Kluwer Academic Publishers, Norwell, MA, USA, 1998.
- [15] Gian Luca Foresti, Petri Mahonen, and Carlo S. Regazzoni, editors. *Multimedia Video-Based Surveillance Systems: Requirements, Issues and Solutions*. Kluwer Academic Publishers, Norwell, MA, USA, 2000.
- [16] C. Alippi, E. Casagrande, F. Scotti, and V. Piuri. Composite real-time image processing for railways track profile measurement. *IEEE Transactions on Instrumentation and Measurement*, 49(3):559–564, June 2000. 0018-9456.
- [17] U. Urfer. Integration of systems and services in central monitoring stations (cms). In *Security Technology, 1995. Proceedings. Institute of Electrical and Electronics Engineers 29th Annual 1995 International Carnahan Conference on*, pages 343–350, 1995.
- [18] A.F. Toal and H. Buxton. Spatio-temporal reasoning within a traffic surveillance system. In G. Sandini, editor, *ECCV92*, volume 588 of *Lecture Notes in Computer Science*, pages 884–892. Springer Berlin Heidelberg, 1992.
- [19] C.P.K. Sherwood. Traffic surveillance and control systems for the tsing ma control area hong kong. In *Road Transport Information and Control, 1998. 9th International Conference on (Conf. Publ. No. 454)*, pages 191–195, 1998.
- [20] D.A. Pritchard. System overview and applications of a panoramic imaging perimeter sensor. In *Security Technology, 1995. Proceedings. Insti-*

- tute of Electrical and Electronics Engineers 29th Annual 1995 International Carnahan Conference on*, pages 420–425, 1995.
- [21] Tom Vercauteren, Aymeric Perchant, Grégoire Malandain, Xavier Pennec, and Nicholas Ayache. Robust Mosaicing with Correction of Motion Distortions and Tissue Deformation for In Vivo Fibered Microscopy. *Medical Image Analysis*, 10(5):673–692, 2006.
  - [22] Peter M. Roth and Martin Winter. Survey of Appearance-Based methods for object recognition. Technical report, Institute for Computer Graphics and Vision, Graz University of Technology, 2008.
  - [23] David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, pages 1150–, Washington, DC, USA, 1999. IEEE Computer Society.
  - [24] Luca Zappella, Xavier Lladó, and Joaquim Salvi. Motion segmentation: a review. In *Proceedings of the 2008 conference on Artificial Intelligence Research and Development: Proceedings of the 11th International Conference of the Catalan Association for Artificial Intelligence*, pages 398–407, Amsterdam, The Netherlands, The Netherlands, 2008. IOS Press.
  - [25] T Bouwmans. Recent advanced statistical background modeling for foreground detection: A systematic survey. *Recent Patents on Computer Science*, 4(3), September 2011.
  - [26] C Stauffer and W E L Grimson. Adaptive background mixture models for real-time tracking. *Proceedings 1999 IEEE Computer Society Con-*

- ference on Computer Vision and Pattern Recognition Cat No PR00149*, 2(c):246–252, 1999.
- [27] N.M. Oliver, B. Rosario, and A.P. Pentland. A bayesian computer vision system for modeling human interactions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):831–843, 2000.
  - [28] Y. Sheikh and M. Shah. Bayesian modeling of dynamic scenes for object detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(11):1778–1792, 2005.
  - [29] A. Elgammal, R. Duraiswami, D. Harwood, and L.S. Davis. Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. *Proceedings of the IEEE*, 90(7):1151–1163, 2002.
  - [30] N. J. B. McFarlane and C. P. Schofield. Segmentation and tracking of piglets in images. *Machine Vision and Applications*, 8(3):187–193, May 1995.
  - [31] Rita Cucchiara, Costantino Grana, Massimo Piccardi, and Andrea Prati. Detecting moving objects, ghosts, and shadows in video streams. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(10):1337–1342, 2003.
  - [32] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
  - [33] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 2, pages 28–31 Vol.2, 2004.

- [34] C.R. Wren, A. Azarbayejani, T. Darrell, and A.P. Pentland. Pfinder: real-time tracking of the human body. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):780–785, 1997.
- [35] Background subtraction online benchmarking. <http://changedetection.net/>.
- [36] Railroad dataset. <http://www.cs.cmu.edu/~yaser/Background.zip/>.
- [37] Yaser Sheikh and Mubarak Shah. Bayesian modeling of dynamic scenes for object detection. *PAMI*, 27:1778–1792, 2005.
- [38] V. Kolmogorov and R. Zabini. What energy functions can be minimized via graph cuts? *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(2):147–159, 2004.
- [39] V. Vineet and P. J. Narayanan. CUDA cuts: Fast graph cuts on the GPU. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW &#039;08. IEEE Computer Society Conference on*, volume 0, pages 1–8, Los Alamitos, CA, USA, June 2008. IEEE.
- [40] Yannick Benezeth, Pierre-Marc Jodoin, Bruno Emile, He?le?ne Laurent, and Christophe Rosenberger. Comparative study of background subtraction algorithms. *Journal of Electronic Imaging*, (3):033003, 2010.
- [41] Peter Hall and M.P. Wand. On the accuracy of binned kernel density estimators. *Journal of Multivariate Analysis*, 56(2):165 – 184, 1996.
- [42] John MacCormick and Andrew Blake. A probabilistic exclusion principle for tracking multiple objects. 2000.



- [43] Vasilis Papadourakis and Antonis Argyros. Multiple objects tracking in the presence of long-term occlusions. *Comput. Vis. Image Underst.*, 114(7):835–846, July 2010.
- [44] B. Bose, Xiaogang Wang, and E. Grimson. Multi-class object tracking algorithm that handles fragmentation and grouping. pages 1 –8, june 2007.
- [45] Antonis A. Argyros and Manolis I. A. Lourakis. Real-time tracking of multiple skin-colored objects with a possibly moving camera. In *In: ECCV*, pages 368–379, 2004.
- [46] Zia Khan, Tucker Balch, and Frank Dellaert. An mcmc-based particle filter for tracking multiple interacting targets. In *ECCV*, 2004.
- [47] Josephine Sullivan and Stefan Carlsson. Tracking and labelling of interacting multiple targets. In *Proceedings of the 9th European conference on Computer Vision - Volume Part III*, ECCV’06, pages 619–632, Berlin, Heidelberg, 2006. Springer-Verlag.
- [48] Qian Yu and G. Medioni. Multiple-target tracking by spatiotemporal monte carlo markov chain data association. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(12):2196–2210, 2009.
- [49] Pierre F. Gabriel, Jacques G. Verly, Justus H. Piater, and André Genon. The state of the art in multiple object tracking under occlusion in video sequences. In *In Advanced Concepts for Intelligent Vision Systems (ACIVS), 2003*, pages 166–173, 2003.

- [50] Stephen J. Mckenna, Sumer Jabri, Zoran Duric, Harry Wechsler, and Azriel Rosenfeld. Tracking groups of people. *Computer Vision and Image Understanding*, 80:42–56, 2000.
- [51] Francois Bremond and Monique Thonnat. Tracking multiple nonrigid objects in video sequences. *IEEE Transactions on Circuits and Systems*, 8(5):585–591, 1998.
- [52] Ismail Haritaoglu, David Harwood, and Larry S Davis. A real time system for detecting and tracking people. *Gesture*, 21/2d:222–227, 1998.
- [53] Sohaib Khan and Mubarak Shah. Tracking people in presence of occlusion. In *In Asian Conference on Computer Vision*, pages 1132–1137, 2000.
- [54] Ismail Haritaoglu, David Harwood, and Larry S Davis. Hydra: multiple people detection and tracking using silhouettes. *Visual Surveillance IEEE Workshop on*, 20742:6–13, 1999.
- [55] Thierry Bouwmans, Laboratoire Mia, Université De La Rochelle, Avenue M Crépeau, and La Rochelle. Recent advanced statistical background modeling for foreground detection - a systematic survey. *Handbook of Pattern Recognition and Computer*, 4(3):147–176, 2010.
- [56] Keni Bernardin, Er Elbs, and Rainer Stiefelhagen. Multiple object tracking performance metrics and evaluation in a smart room environment, 2006.
- [57] Peter Hall and M P Wand. On the accuracy of binned kernel density estimators. *Journal of Multivariate Analysis*, 56(2):165–184, 1996.

- [58] Matthew Brown and David G. Lowe. Automatic panoramic image stitching using invariant features. *Int. J. Comput. Vision*, 74(1):59–73, August 2007.
- [59] Tae Eun Choe, Isaac Cohen, Munwai Lee, and Gerard Medioni. Optimal global mosaic generation from retinal images. In *Proceedings of the 18th International Conference on Pattern Recognition - Volume 03*, ICPR '06, pages 681–684, Washington, DC, USA, 2006. IEEE Computer Society.
- [60] Ludovico Carozza, Alessandro Bevilacqua, and Filippo Piccinini. Mosaicing of optical microscope imagery based on visual information. In *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pages 6162–6165. IEEE, 2011.
- [61] Heeseung Choi, Kyoungtaek Choi, and Jaihie Kim. Mosaicing touchless and mirror-reflected fingerprint images. *Information Forensics and Security, IEEE Transactions on*, 5(1):52–61, 2010.
- [62] Yuping Lin and G. Medioni. Map-enhanced uav image sequence registration and synchronization of multiple image sequences. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–7, 2007.
- [63] R. Prados, R. Garcia, N. Gracias, J. Escartin, and L. Neumann. A novel blending technique for underwater gigamosaicing. *Oceanic Engineering, IEEE Journal of*, 37(4):626–644, 2012.
- [64] Video compression using mosaic representations. *Signal Processing: Image Communication*, 7.

- [65] Richard Szeliski. Image alignment and stitching: a tutorial. *Found. Trends. Comput. Graph. Vis.*, 2(1):1–104, January 2006.
- [66] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2*, IJCAI’81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.
- [67] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. *Int. J. Comput. Vision*, 56(3):221–255, February 2004.
- [68] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008.
- [69] Tinne Tuytelaars and Luc J. Van Gool. Matching widely separated views based on affine invariant regions. *International Journal of Computer Vision*, 59(1):61–85, 2004.
- [70] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(10):1615–1630, oct. 2005.
- [71] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.
- [72] Steven Lovegrove and Andrew J. Davison. Real-time spherical mosaicing using whole image alignment. In *Proceedings of the 11th European*

- conference on computer vision conference on Computer vision: Part III*, ECCV'10, pages 73–86, Berlin, Heidelberg, 2010. Springer-Verlag.
- [73] Michal Irani and P. Anandan. Parallax geometry of pairs of points for 3d scene analysis. In *Proceedings of the 4th European Conference on Computer Vision-Volume I - Volume I*, ECCV '96, pages 17–30, London, UK, UK, 1996. Springer-Verlag.
  - [74] R. Kumar, P. Anandan, and K. Hanna. Direct recovery of shape from multiple views: a parallax based approach. In *Pattern Recognition, 1994. Vol. 1 - Conference A: Computer Vision and Image Processing, Proceedings of the 12th IAPR International Conference on*, volume 1, pages 685 –688 vol.1, oct 1994.
  - [75] M. Irani, P. Anandan, and M. Cohen. Direct recovery of planar-parallax from multiple frames. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(11):1528 – 1534, nov 2002.
  - [76] T. Templeton, D.H. Shim, C. Geyer, and S.S. Sastry. Autonomous vision-based landing and terrain mapping using an mpc-controlled unmanned rotorcraft. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 1349–1356, 2007.
  - [77] Chang Yuan, Gerard Medioni, Jinman Kang, and Isaac Cohen. Detecting motion regions in the presence of a strong parallax from a moving camera by multiview geometric constraints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(9):1627–1641, 2007.

- [78] Alex Rav-Acha, Giora Engel, and Shmuel Peleg. Minimal aspect distortion (mad) mosaicing of long scenes. *Int. J. Comput. Vision*, 78(2-3):187–206, July 2008.
- [79] Qi Zhi and Jeremy R. Cooperstock. Toward dynamic image mosaic generation with robustness to parallax. *Trans. Img. Proc.*, 21(1):366–378, January 2012.
- [80] Wen-Yan Lin, Siying Liu, Y. Matsushita, Tian-Tsong Ng, and Loong-Fah Cheong. Smoothly varying affine stitching. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, pages 345–352, Washington, DC, USA, 2011. IEEE Computer Society.
- [81] Barbara Zitová and Jan Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21:977–1000, 2003.
- [82] J.A. Little, D. L G Hill, and D.J. Hawkes. Deformations incorporating rigid structures [medical imaging]. In *Mathematical Methods in Biomedical Image Analysis, 1996., Proceedings of the Workshop on*, pages 104–113, 1996.
- [83] Javier Civera, Andrew J. Davison, Juan A. Magallón, and J. M. Montiel. Drift-free real-time sequential mosaicing. *Int. J. Comput. Vision*, 81(2):128–137, 2009.
- [84] Eun-Young Kang, I. Cohen, and G. Medioni. A graph-based global registration for 2d mosaics. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 1, pages 257–260 vol.1, 2000.

- [85] Yuping Lin and Gerard Medioni. Map-enhanced uav image sequence registration and synchronization of multiple image sequences. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 0:1–7, 2007.
- [86] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, ICCV '99, pages 298–372, London, UK, 2000. Springer-Verlag.
- [87] T. Suzuki, Y. Amano, and T. Hashizume. Vision based localization of a small uav for generating a large mosaic image. In *SICE Annual Conference 2010, Proceedings of*, pages 2960–2964, 2010.
- [88] D. Blake Barber, Joshua D. Redding, Timothy W. Mclain, Randal W. Beard, and Clark N. Taylor. Vision-based target geo-location using a fixed-wing miniature air vehicle. *J. Intell. Robotics Syst.*, 47(4):361–382, December 2006.
- [89] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.
- [90] P. H. S. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78:2000, 2000.
- [91] Ben Tordoff and David W. Murray. Guided sampling and consensus for motion estimation. In *Proceedings of the 7th European Conference on Computer Vision-Part I*, ECCV '02, pages 82–98, London, UK, UK, 2002. Springer-Verlag.

- [92] Philip J. Schneider and David Eberly. *Geometric Tools for Computer Graphics*. Elsevier Science Inc., New York, NY, USA, 2002.
- [93] Peter J. Burt and Edward H. Adelson. A multiresolution spline with application to image mosaics. *ACM Trans. Graph.*, 2(4):217–236, October 1983.
- [94] AerialRobotics Dataset. <ftp://www.aerialrobotics.eu/>.
- [95] UAVPeople Dataset. <http://www.uavpeople.com/download/dataset>.